

AD-A040 588

GAERTNER (W W) RESEARCH INC STAMFORD CONN  
DESIGN, CONSTRUCTION AND INSTALLATION OF DATA MANIPULATOR.(U)

F/G 9/2

MAY 77 W W GAERTNER, D T LEE, C T RETTER

F30602-75-C-0290

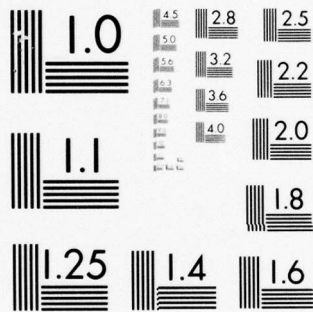
UNCLASSIFIED

RADC-TR-77-166

NL

1 OF 1  
AD  
A040588





MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

AD A 040588

RADC-TR-77-166  
Final Technical Report  
May 1977

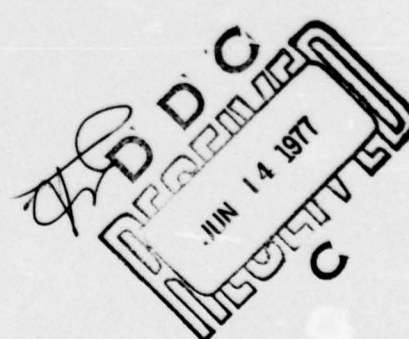
12  
B.S.



DESIGN, CONSTRUCTION AND INSTALLATION OF DATA MANIPULATOR

W. W. Gaertner Research Inc

Approved for public release; distribution unlimited.



AD No. \_\_\_\_\_  
DDC FILE COPY

ROME AIR DEVELOPMENT CENTER  
Air Force Systems Command  
Griffiss Air Force Base, New York 13441

This report has been reviewed and is approved for publication.

ROBERT D. KRUTZ, Colonel, USAF  
Chief, Information Sciences Division

JOHN P. HUSS  
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

AIR MAIL  
RHS  
ONE  
DRAIN OUTLET  
AIRSHIP

WIRE SOLDER  
BUT SOLDER

✓  
☒  
☐

BY AIRMAIL SPECIAL DELIVERY

ALL MAIL BY AIR SPECIAL

P



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-77-166	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DESIGN, CONSTRUCTION AND INSTALLATION OF DATA MANIPULATOR.	5. TYPE OF REPORT & PERIOD COVERED Final Technical Report.	6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) W. W. Gaertner, D. T. W. Lee, C. T. Retter, I. M. Singh	8. CONTRACT OR GRANT NUMBER(s) F30602-75-C-0290	9. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 5550126
10. PERFORMING ORGANIZATION NAME AND ADDRESS W. W. Gaertner Research Inc 205 Saddle Hill Road Stamford CT 06903	11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (ISCA) Griffiss AFB NY 13441	12. REPORT DATE May 1977
13. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same	14. SECURITY CLASS. (of this report) UNCLASSIFIED	15. NUMBER OF PAGES 80
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Same		
18. SUPPLEMENTARY NOTES RADC Project Engineer: Murray Kesselman (ISCA)		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Data manipulation, cross-point switch, parallel processing, associative processing		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report describes the design, fabrication and installation of a Data Manipulator as originally suggested by Tse-yun Feng in RADC Technical Report TR-73-292, "The Design of a Versatile Line Manipulator". The function of a Data Manipulator, namely that of a fully programmable cross-point switch with masked inputs and outputs is explained first, including its various uses to provide a versatile communication capability between the different building blocks of a computer architecture, with particular emphasis on parallel and associative computer systems. Two major design alternatives are then discussed.		

DD FORM 1473

1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

388 544

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

one relying primarily on a CMOS custom-LSI gate-array chip, the other based on a standard off-the-shelf bipolar multiplexer with on-chip decoding. The various considerations which led to the selection of the second approach are described.

The Target Specifications to develop a Data Manipulator to specifically operate with the RADC STARAN Computer system are presented next. While initially only a 64x64 line capability was specified, the actual design allows expansion to a full 256x256-line system simply by duplicating already designed card types, without the need for redesign to accommodate the additional addressability and fan out.

A detailed description of the System Design, including the interfaces to the STARAN Computer and the incorporation of a built-in Self-Test Computer (a PDP-11/03 with 8K of memory) into the Data Manipulator are described next. The functional partitioning of the overall equipment architecture, the circuit design and operation of the individual circuit boards are described. Specifically, the entire Data-Manipulator function is implemented by two major types of circuit boards, namely the Register Card which holds all the registers (Address Control Registers, Input Control Register, Output Control Register, Input Mask Register, Output Mask Register, Input Data Register and Output Data Register) required to set all switches in the Data Manipulator and provide input and output masking as well as buffering of input and output data and the Line Select Card which holds all the multiplexers and decoders necessary to provide the cross-point switching function.

Chapter 3.5 is devoted to the Data-Manipulator instruction repertoire and other software, required to operate the Data Manipulator under the control of the STARAN PIO Control facility.

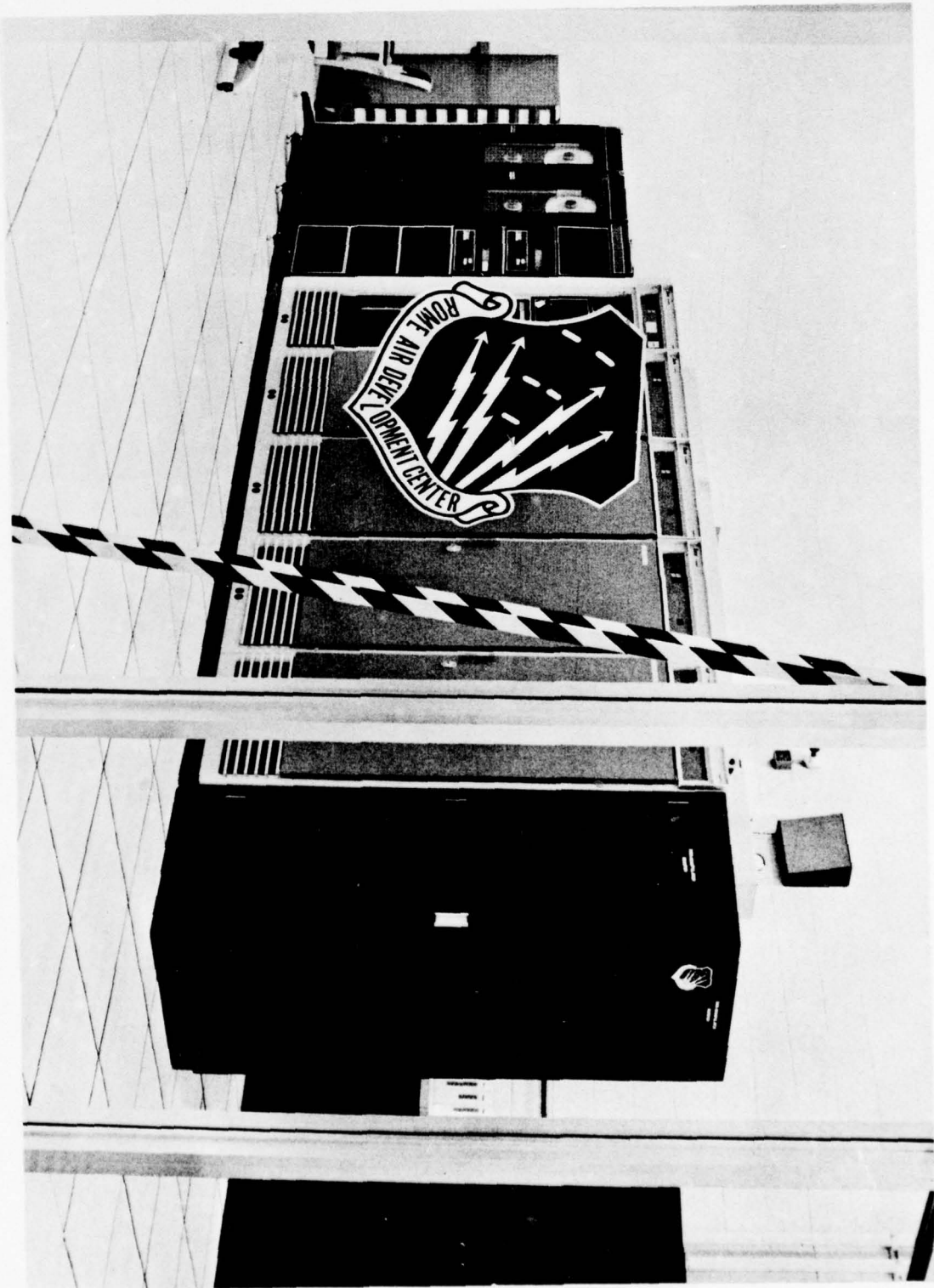
The physical construction of the Data Manipulator is explained and illustrated in Chapter 4.

The use of a built-in Self-Test Computer for initial development, debugging, installation and maintenance of the equipment is described in detail in Chapter 5. It allows one to perform over two million verified manipulations per hour, systematically chosen to test all modes of all circuits in the system, whether the Data Manipulator is connected to a STARAN Computer or not, allowing substantial life-test and burn-in operation prior to delivery.

The installation, integration and operation of the Data Manipulator with the RADC STARAN Computer is described, and the conclusion is drawn that a Data Manipulator not only provides a very useful computer-system capability but can also readily be designed and constructed as an add-on to existing computer or communication systems.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)



Data Manipulator installed at RADC Parallel/Associative Computer Facility



## TABLE OF CONTENTS

Chapter No.	Page No.
LIST OF REFERENCED DRAWINGS	iii
1. INTRODUCTION	1
2. THE FUNCTION OF A DATA MANIPULATOR	3
3. DESIGN AND DEVELOPMENT OF A DATA MANIPULATOR	12
3.1 Target Specifications	12
3.2 Design Alternatives and Selection of Preferred Approach	13
3.3 Description of System Design	25
3.4 Hardware Design and Operation	29
3.4.1 Functional System Partitioning	29
3.4.2 STARAN Interface	29
3.4.3 Line Select Cards	30
3.4.4 Register Cards	30
3.4.5 ICR/OCR to ACR Conversion	32
3.4.6 Control Card	34
3.4.7 Modes of Operation of the Data Manipulator	36
3.4.8 LSI-11 Interface Card	37
3.4.9 Control and Status Register (CSR)	38
3.5 Data Manipulator Software	40
4. PHYSICAL CONSTRUCTION	44
5. SELF-TEST AND MAINTAINABILITY	54
5.1 Description of the Self-Test Program	54
5.1.1 Test Routines	54
5.2 Running the Data Manipulator Self-Test Program	57
5.2.1 Alternate Entry Points	57
5.2.2 Format of Diagnostic Messages	58
5.2.3 Error Maps	59
5.2.4 Description of Diagnostic Messages	59
5.3 Debugging Aids	74
6. INSTALLATION AND OPERATION OF THE DATA MANIPULATOR IN THE RADC PARALLEL/ASSOCIATIVE COMPUTER FACILITY	77
7. CONCLUSIONS AND OUTLOOK	80

## LIST OF ILLUSTRATIONS AND TABLES

No.	Title	Page
2-1	Function of a general Data Manipulator	4
2-2	Some common operations performed by a Data Manipulator	5
2-3	List of major data manipulating functions	6
2-4	Data Manipulator with Input and Output Control Registers (ICR and OCR)	7
2-5	Data Manipulators can be incorporated into computer architectures in various locations	9
2-6	A Data Manipulator can be used to provide communication capability between Processing Elements in a parallel computer system	11
3.2-1	A versatile line manipulator (VLM) organization suggested by T. Feng	15
3.2-2	The logic circuit of BLMC (Basic Line Manipulator Circuit) Cell (i,j) as suggested by T. Feng	16
3.2-3	CMOS design of BLMC cell requiring 10 transistor pairs	17
3.2-4	CMOS design for 3-to-8 decoder with chip enable requiring 35 transistor pairs	18
3.2-5	Expansion from 32x32 to 256x256 line array	21
3.2-6	Implementation of DM function by use of tri-state 16-to-1 multiplexer ICs.	23
3.3-1	Block Diagram of Data Manipulator	26
3.4.4-1	Block Diagram of Register Card	31
3.4.5-1	ICR/OCR to ACR Conversion	33
3.4.5-2	Priority Arbiter	35
3.5-1	STARAN Instructions for Data Manipulator	41
4.0-1	The RADC STARAN facility with Data Manipulator Installed	45
4.0-2	Data Manipulator with cabinet door open	46
4.0-3	Data Manipulator Back Plane	47
4.0-4	Register Card	48
4.0-5	Line Select Card	50
4.0-6	STARAN Interface Card - Input	51
4.0-7	STARAN Interface Card - Output	52
4.0-8	STARAN Interface Card - Control	53
5.2.4-1(a)	Example of ACR Error Map (MSG #14)	62
5.2.4-1(b)	Example of ACR Error Map (MSG #14)	63
5.2.4-2(a)	Example of Register Error Map (MSG #15)	65
5.2.4-2(b)	Example of Register Error Map (MSG #15)	66
5.2.4-3(a)	Example of Data Manipulation Error Map	69
5.2.4-3(b)	Example of Data Manipulation Error Map	70
5.2.4-4	Example of ICR/OCR Conversion Error display	72
5.2.4-5	Example of ICR Modification error display	73
5.3-1	Format of DM Register Display	75



## LIST OF REFERENCED DRAWINGS

All drawings other than the figures in this report are contained in a separate D-sized file (Data Manipulator - Engineering Drawings). Its table of contents is reproduced below.

### DATA MANIPULATOR - ENGINEERING DRAWINGS

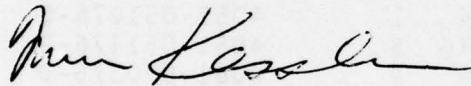
#### TABLE OF CONTENTS

No.	Title	Pages	Drawing No.
1	DM CONTROL INTERFACE	1	4051-011576-D
2	DM INPUT INTERFACE	2	4051-011676-D
3	DM OUTPUT INTERFACE	2	4051-012076-D
4	DM REGISTER CARD - BLOCK DIAGRAM	1	4051-051076-D
5	DM REGISTER CARD - LOGIC DIAGRAM	5	4051-051476-D
6	DM LINE SELECT CARD	9	4051-070876-D
7	DM CONTROL CARD	4	4051-052076-D
8	DM LSI-11 INTERFACE CARD	2	4051-052776-D
9	DM TEST COMPUTER ADDRESS MAP	1	4051-070676-D
10	CARD CONFIGURATION IN DM CABINET	1	4051-052876-D
11	POWER BAR	1	4051-072776-D1
12	GROUND BAR	1	4051-072776-D2
13	FRONT PANEL	1	4051-072776-D3
14	POWER SUPPLY SUPPORT BARS (SIDE BARS)	1	4051-072876-C1
15	POWER SUPPLY SUPPORT BARS	1	4051-072876-C2
16	POWER SUPPLY SUPPORT BARS	1	4051-072876-C3
17	CARD GUIDE	1	4051-072976-C1
18	CARD GUIDE SUPPORT	1	4051-072976-C2
19	CENTER STIFFENER	1	4051-072976-C3
20	BACK PANEL	1	4051-072976-D4
21	FRONT STIFFENER	1	4051-073176-C1
22	BACK EDGE STIFFENER	1	4051-081276-C
23	SUPPORT BARS FOR DATA MANIPULATOR BACKPLANE	1	4051-071676-D
24	WIRING DIAGRAM FOR STARAN DATA MANIPULATOR RACK	1	4051-081976-C

## EVALUATION

Successful application of parallel processing to useful problems depends heavily on developing methods to support intercommunications between the parallel processing elements. The Data Manipulator represents one of the most advanced techniques for supporting this intercommunication. It is capable of providing all interconnect patterns such as shifts, expand replicate, and shuffle.

This capability when interfaced to the STARAN Associative Processor provides one more tool that is useful in exploring the application of parallel processing to Air Force computational problems as related in TPO-V A.



MURRAY KESSELMAN  
Project Engineer

## 1. INTRODUCTION

It is by now well recognized that most parallel processing systems require a versatile communication facility to pass data (a) between the different processing elements, (b) between the processing elements as a group and the memory modules and (c) between the input/output devices and memory and/or processing elements. This communication capability is used to exchange data (such as interim results) between processing elements, to rearrange data in memory, and even to rearrange bits within the same parallel word. The term Data Manipulation has been introduced to define this function which must be considered a computer-system building block of equal importance to the processing element array, the high speed memory, the mass storage and the system control. Surprisingly, this fact has not been recognized adequately in the design of several existing parallel-processor systems, with the result that the execution of many important algorithms is slowed down significantly.

This report describes the design, construction, installation and performance of a Data Manipulator which, while integrated specifically into a STARAN computer system, is applicable to any other parallel computer architecture.

The report is organized as follows:

Chapter 2 describes the functions performed by a Data Manipulator and how it fits into the overall computer architecture.

Chapter 3 deals with the design and development of the Data Manipulator, starting with the Target Specifications, followed by a discussion of Design Alternatives and the Selection of the Preferred Approach, including a justification of the choice. Descriptions of the System Design, the Hardware Design and the Operation of the Data Manipulator follow. Chapter 3.5 contains a discussion of the Data-Manipulator instruction repertoire and other software.

Chapter 4 contains a discussion and illustrations of the physical construction of the equipment.

Details of the Self-Test capability and of the Maintainability of the Data Manipulator are presented in Chapter 5. Aside from documenting the extensive design-for-maintainability features of the Data Manipulator it also provides additional insight into the general operation of the equipment.

Chapter 6 briefly recounts the installation and operation of the Data Manipulator at the RADC Parallel/Associative Computer Facility, and Chapter 7 presents brief Conclusions and Outlook.



## 2. THE FUNCTION OF A DATA MANIPULATOR

Figure 2-1 illustrates the function of a general Data Manipulator: The data in any bit location of the input data register can be routed to any bit location in the output data register. Both inputs and outputs can be masked with the effect that any line with a logical 1 in the mask register is allowed to pass information, whereas information on any line with a logical 0 is blocked. This is illustrated in Figure 2-1 where a heavy line indicates a path through which information flows, whereas a light line indicates a blocked path.

The Data Manipulator described in this report is unidirectional, i.e. the information always only flows from input to output.

There are obviously many different ways for specifying the internal routing in the Data Manipulator. One of the most effective is to assign an Address Control Register (ACR) to each output line and have this ACR specify the address of the input line which is to serve as the source of information for the particular output line. In a 256x256-line Data Manipulator the Address Control Register (ACR) for each output line contains 8 bits. The reason for placing the ACRs on the output rather than the input is that this arrangement allows one to select a single input line as the source for multiple output lines, a function which is very useful for replicating, multiplying and otherwise fanning-out of information from a single input line.

Figure 2-2 illustrates some common operations performed by a Data Manipulator and Figure 2-3 lists most major data manipulating functions in a systematic manner (see Tse-yun Feng, "The Design of a Versatile Line Manipulator", RADC Contract F30602-72-C-0281, Technical Report TR-73-292, June 1973).

While the use of the Address Control Registers (ACRs) allows the implementation of any arbitrary unidirectional routing pattern, it has been considered desirable to allow the realization of a specific subset of routing patterns through the use of an Input Control Register (ICR) and an Output Control Register (OCR), as shown in Figure 2-4. The ICR and the OCR are loaded with a pattern of logical



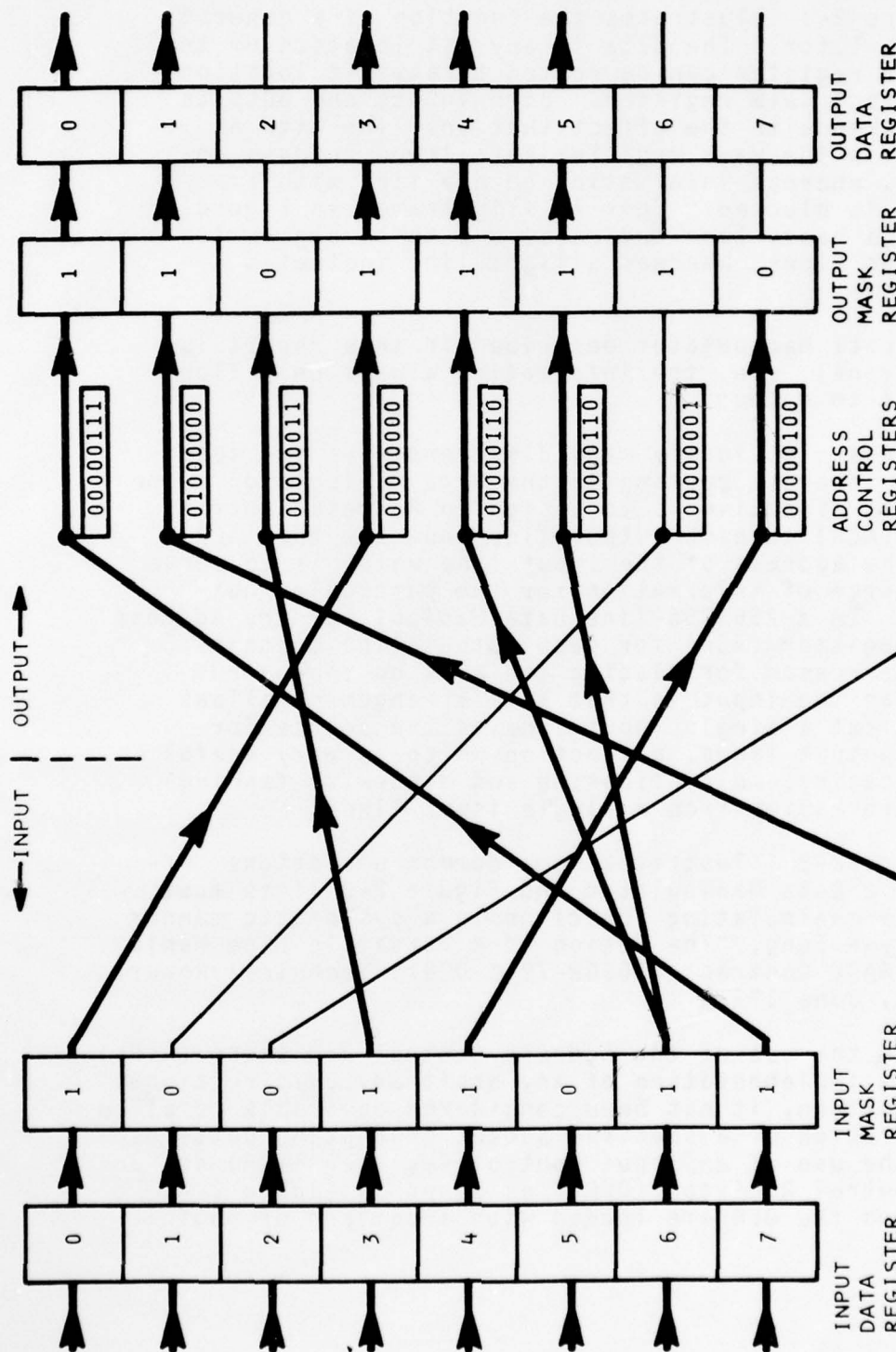
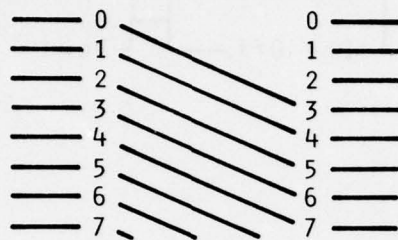
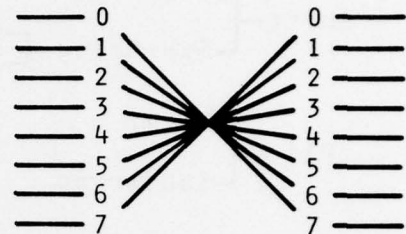


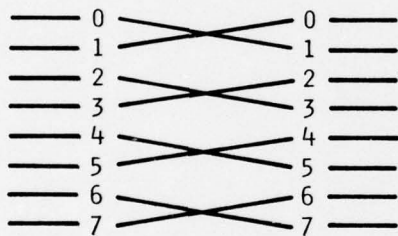
Figure 2-1. Function of a general Data Manipulator



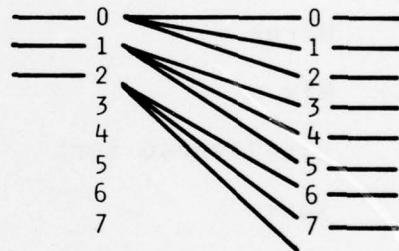
SHIFT



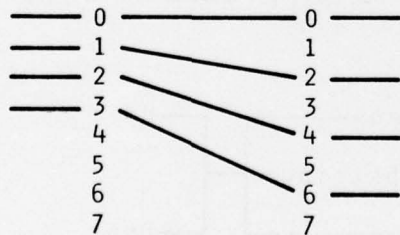
FLIP



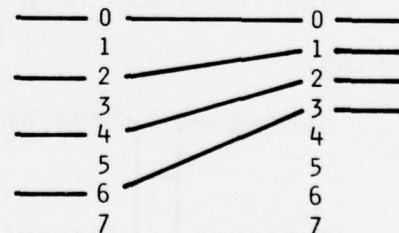
TRANSPOSE



MULTIPLICATE



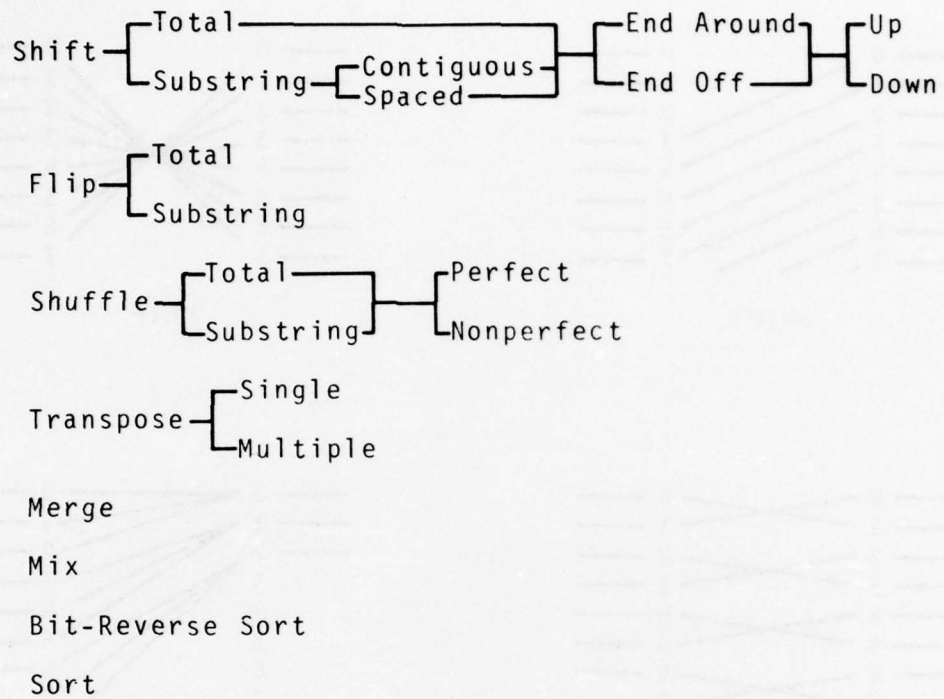
SPREAD



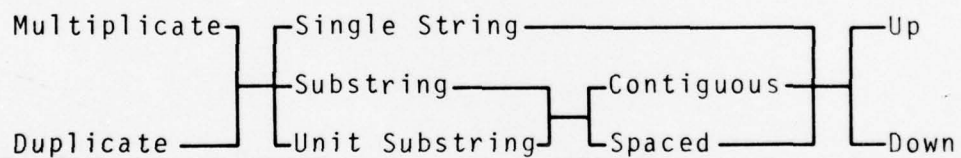
COMPRESS

Figure 2-2. Some common operations performed by a Data Manipulator.

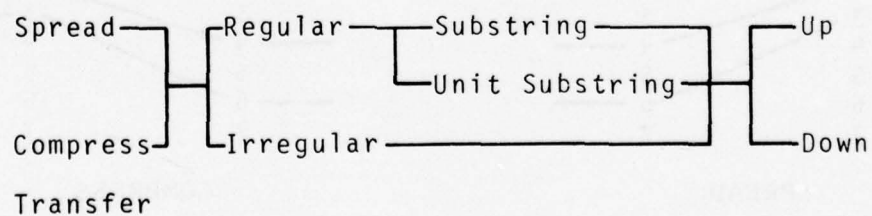
### PERMUTING



### REPLICATING



### SPACING



### MASKING

Figure 2-3. List of major data manipulating functions.

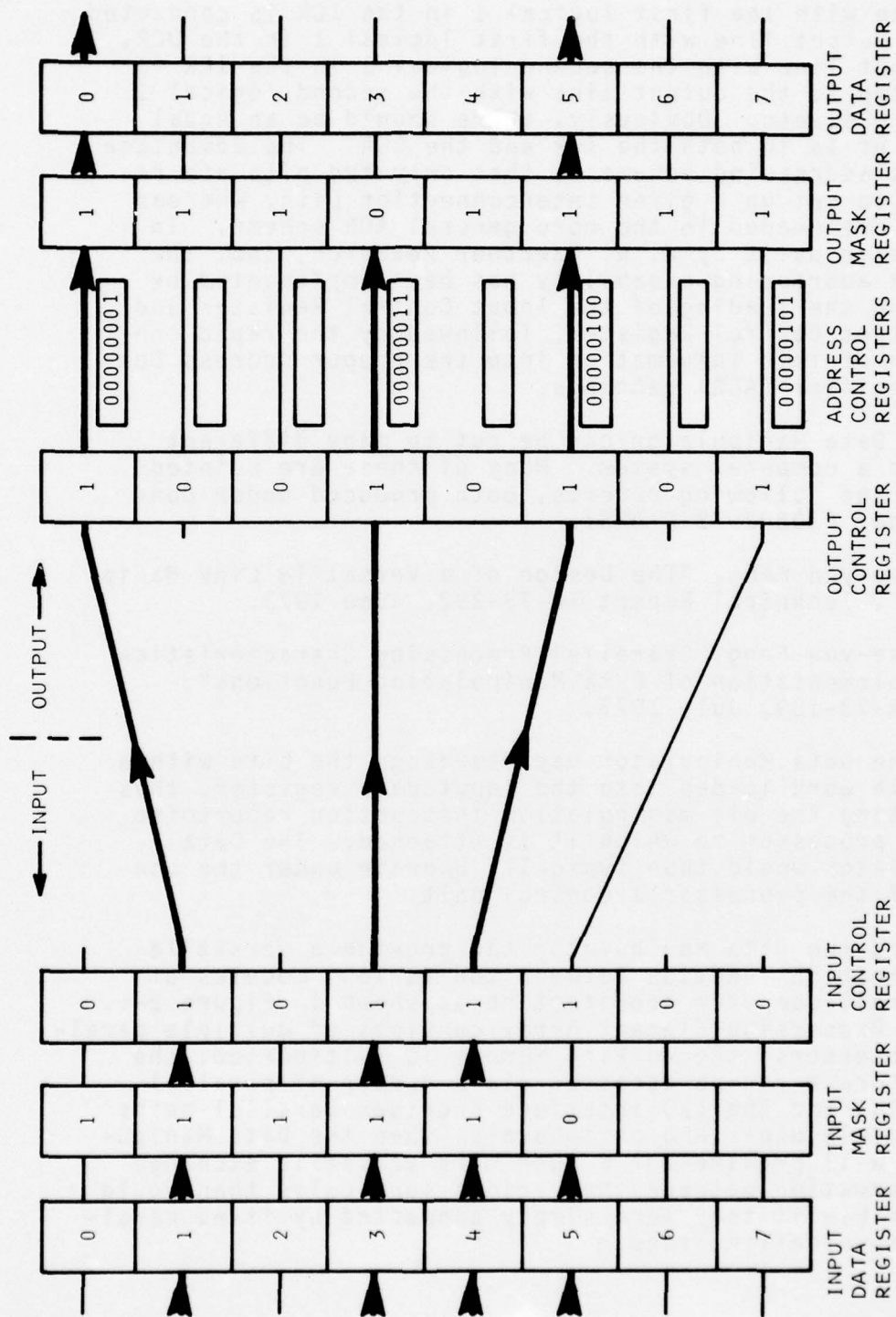


Figure 2-4. Data Manipulator with Input and Output Control Registers (ICR and OCR).



0s and 1s, and the routing is arranged such that the input line with the first logical 1 in the ICR is connected to the output line with the first logical 1 in the OCR, the input line with the second logical 1 in the ICR is connected to the output line with the second logical 1 in the OCR, etc. Obviously, there should be an equal number of 1s in both the ICR and the OCR. The advantage of this addressing scheme is that only two bits are required to set up a given interconnection path, whereas 8 bits are needed in the more general ACR scheme. In the design built by W. W. Gaertner Research, Inc. the ICR/OCR addressing capability has been implemented by allowing the loading of the Input Control Register and the Output Control Register, followed by the rapid conversion of this information into the proper Address Control Register (ACR) settings.

A Data Manipulator can be put to many different uses in a computer system. Many of these are pointed out in the following reports, both produced under Contract no. F30602-72-C-0281:

Tse-yun Feng, "The Design of a Versatile Line Manipulator", Technical Report TR-73-292, June 1973.

Tse-yun Feng, "Parallel Processing Characteristics and Implementation of Data Manipulating Functions", RADC-TR-73-189, July 1973.

The Data Manipulator may rearrange the bits within a single word loaded into the input data register, thus augmenting the bit-manipulation instruction repertoire of the processor to which it is attached. The Data Manipulator would then typically operate under the control of the processor's control unit.

Or, the Data Manipulator can provide a versatile communication function between the various modules of an overall computer architecture as shown in Figure 2-5. If the Processing-Element Array consists of multiple parallel processors; the Working Memory is multiported; the Mass Storage can be accessed via a number of parallel channels; and the I/O interface provides parallel paths to multiple users and peripherals, then the Data Manipulators will provide for a much more versatile exchange of information between the various submodules than would be possible if they were simply connected by fixed parallel communication lines.



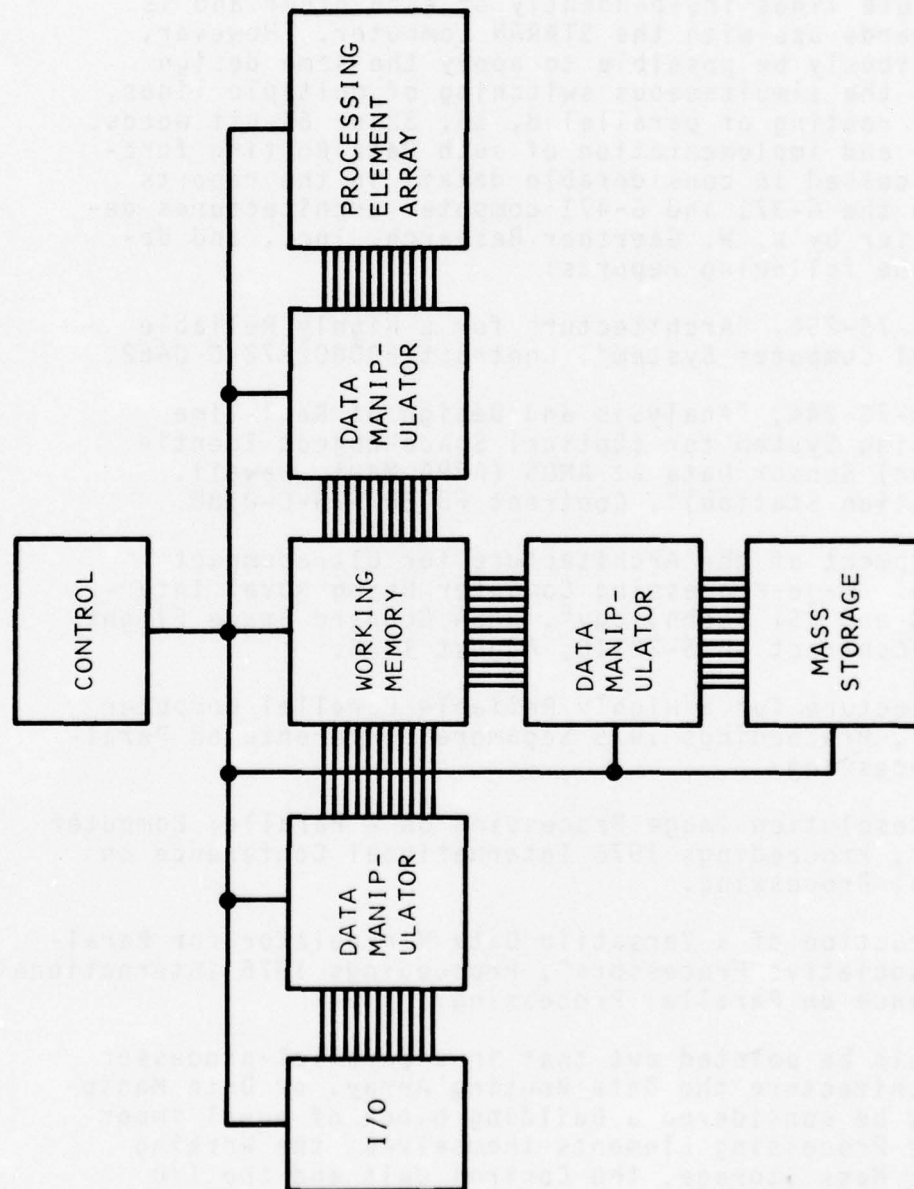


Figure 2-5. Data Manipulators can be incorporated into computer architectures in various locations.

Furthermore, a Data Manipulator can also be used to provide for versatile data routing between the Processing Elements of a parallel computer system, as shown in Figure 2-6.

The Data Manipulator designed under this contract switches single lines independently of each other and is oriented towards use with the STARAN computer. However, it would obviously be possible to apply the same design principle to the simultaneous switching of multiple lines, to allow the routing of parallel 8, 16, 32 or 64-bit words. The need for and implementation of such Data-Routing functions is discussed in considerable detail in the reports dealing with the G-371 and G-471 computer architectures developed earlier by W. W. Gaertner Research, Inc., and described in the following reports:

RADC-TR-75-256, "Architecture for a Highly Reliable Parallel Computer System", Contract F30602-72-C-0462.

RADC-TR-76-244, "Analysis and Design of Real-Time Processing System for (Optical Space-Object Identification) Sensor Data at AMOS (ARPA Maui, Hawaii, Observation Station)", Contract F30602-75-C-0180.

"Development of the Architecture for Ultracompact Parallel Image-Processing Computer Using Novel Interconnect and LSI Technology", NASA Goddard Space Flight Center Contract NAS5-22318, August 1976.

"Architecture for a Highly Reliable Parallel Computer System", Proceedings 1975 Sagamore Conference on Parallel Processing.

"High-Resolution Image Processing on a Parallel Computer System", Proceedings 1976 International Conference on Parallel Processing.

"Construction of a Versatile Data Manipulator for Parallel/Associative Processors", Proceedings 1976 International Conference on Parallel Processing.

It should be pointed out that in a parallel-processor computer architecture the Data Routing Array, or Data Manipulator, must be considered a building block of equal importance to the Processing Elements themselves, the Working Storage, the Mass Storage, the Control Unit and the I/O Interface. Any architecture without this facility is usually severely limited in its ability to handle general-purpose problems.

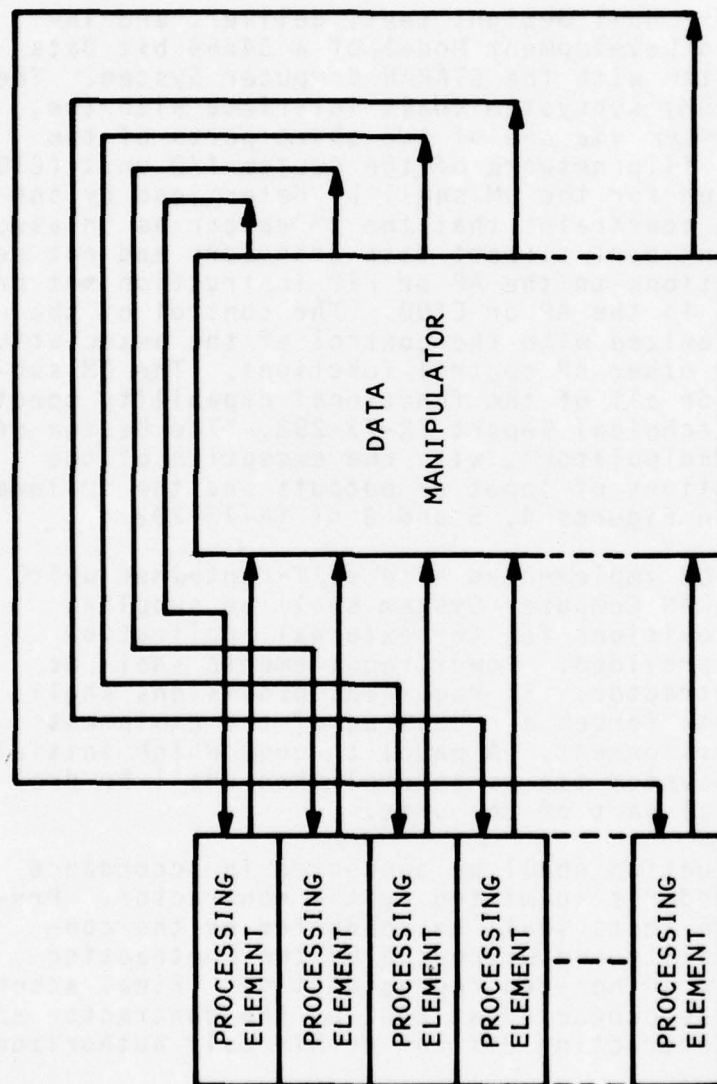


Figure 2-6. A Data Manipulator can be used to provide communication capability between Processing Elements in a parallel computer system.



### 3. DESIGN AND DEVELOPMENT OF DATA MANIPULATOR

#### 3.1 Target Specifications

The Data Manipulator was designed against the following Target Specifications:

The contractor shall design, test, deliver, and integrate an advanced Development Model of a 64x64 bit Data Manipulator subsystem with the STARAN Computer System. The Data Manipulator (DM) subsystem shall interface with the STARAN Computer System via one of the spare ports of the parallel I/O (PIO) flip network of the custom I/O unit (CIOU). The control required for the DM shall be determined by the contractor within the constraint that the DM appear as an associative array during port-to-port data transfers and not require any modifications to the AP or PIO instruction set or control mechanisms in the AP or CIOU. The control of the DM shall be synchronized with the control of the associative processor (AP) and other AP control functions. The DM subsystem shall provide all of the functional capability specified in the RADC Technical Report TR-73-292, "The Design of a Versatile Line Manipulator", with the exception of the complementing functions of input or outputs and the implementation described in Figures 4, 5 and 6 of TR-73-292.

The DM shall be implemented as a self-contained unit. Cabling to the STARAN Computer System shall be supplied with the unit. Provisions for the external application of power shall be provided. Power requirements shall be defined by the contractor. If required, provisions shall be made for adequate forced air cooling of the equipment in a laboratory environment. A panel through which initialization of the subsystem can be accomplished shall be provided as an integral part of the unit.

Test and evaluation shall be conducted in accordance with the test procedures submitted by the contractor. Preliminary acceptance tests shall be conducted by the contractor at his facility and witnessed by the Contracting Officer or his duly authorized representative. Final acceptance tests shall be conducted at RADC by the contractor and witnessed by the Contracting Officer or his duly authorized representative.

The DM subsystem shall be capable of operating within a controlled laboratory environment with an ambient temperature of  $70 \pm 10$  degrees F. Commercial grade components (i.e., components with an 0 degree C to 75 degree C (32

degrees F to 167 degrees F) operating temperature range), with the exception of the LSI devices, and commercial grade packaging shall be acceptable except as otherwise noted herein. The contractor shall be responsible for assuring that all specified quality and reliability assurance requirements for microcircuits have been accomplished. All microcircuits shall be screened in accordance with MIL-STD-883A, Method 5004, Class B. Nonstandard part approval requests shall be submitted for all microcircuits and shall contain the quality and reliability assurance procedures proposed for the microcircuits, in accordance with MIL-STD-454, Requirement 22. Detail specifications shall be in accordance with MIL-M-38510 format to the maximum extent possible, and include both the rework procedures and burn-in test circuit for each microcircuit. All microcircuits used in the equipments shall be packaged in hermetic packages and no "plastic" (i.e., epoxy, silicone, phenolic or other organic materials) encapsulated devices shall be accepted.

The setup time required to establish a data manipulation format using the Input Control Register and Output Control Register mode shall be less than 4 microseconds.

The throughput time for data passing through the DM (from latched input to latched output), once the data manipulation format has been established, shall be less than 750 nanoseconds.

### 3.2 Design Alternatives and Selection of Preferred Approach

Thus, the function to be performed by the Data Manipulator is well defined, namely that of a programmable cross-point switch with masking at input and output. However, there are many alternatives with respect to the

- (a) architecture,
- (b) logic design, and
- (c) technological implementation

which may be chosen for the actual realization of such a Data Manipulator. Out of the large matrix of possibilities we shall discuss two approaches which lie more or less at opposite ends of the spectrum of options.

The first approach relies on a gate array, using custom LSI circuitry. The ACR, ICR and OCR operations are fully integrated and the ICR and OCR decoding is performed on the



chip.

The second alternative uses off-the-shelf 16-to-1 MSI line-select circuits for the basic cross-point switching and ACR-decoding functions, and the ACR mode of operation is functionally separated from the ICR/OCR mode of operation.

Figures 3.2-1 and 3.2-2, from the previously referenced report by Tse-yun Feng show the proposed DM architecture and the implementation of the Basic Line Manipulator Circuit (BLMC) cell. For clarity, in Figure 3.2-1, the register functions have been separated from the decoder and switching functions into a Memory section and a Gate-Array section of the proposed Data Manipulator. This separation is natural and occurs in all Data-Manipulator architectures. The main differences between architectures lies in the implementation of the Gate Array. In the Feng design, a cellular array of the Basic Line Manipulator Circuit is suggested as shown in Figure 3.2-2. A CMOS implementation of the BLMC is given in Figure 3.2-3 indicating a complexity of 20 transistors. The additional decoding circuitry needed is illustrated in Figure 3.2-4. A 3-to-8 decoder with chip enable requires 70 transistors.

In a DM involving  $N \times N$  lines,  $N^2$  crossbar switch gates are needed; addressing via the Address Control Register (ACR) requires  $N \log_2 N$  bits of memory to define the switch positions and  $N$  each  $\log_2 N$ -to- $N$  decoders; addressing via the Input and Output Control Registers (ICR, OCR) requires  $2N$  bits of memory to define the switch positions, and  $4N^2$  to  $5N^2$  gates to decode the memory. The Input and Output Mask Registers (IMR, OMR) consist of  $2N$  bits of memory and  $2N$  AND gates. Thus, the circuit complexity for the implementation of a DM with a given  $N$  is well defined.

The number of gates needed for an  $N \times N$  array are shown in the following table:

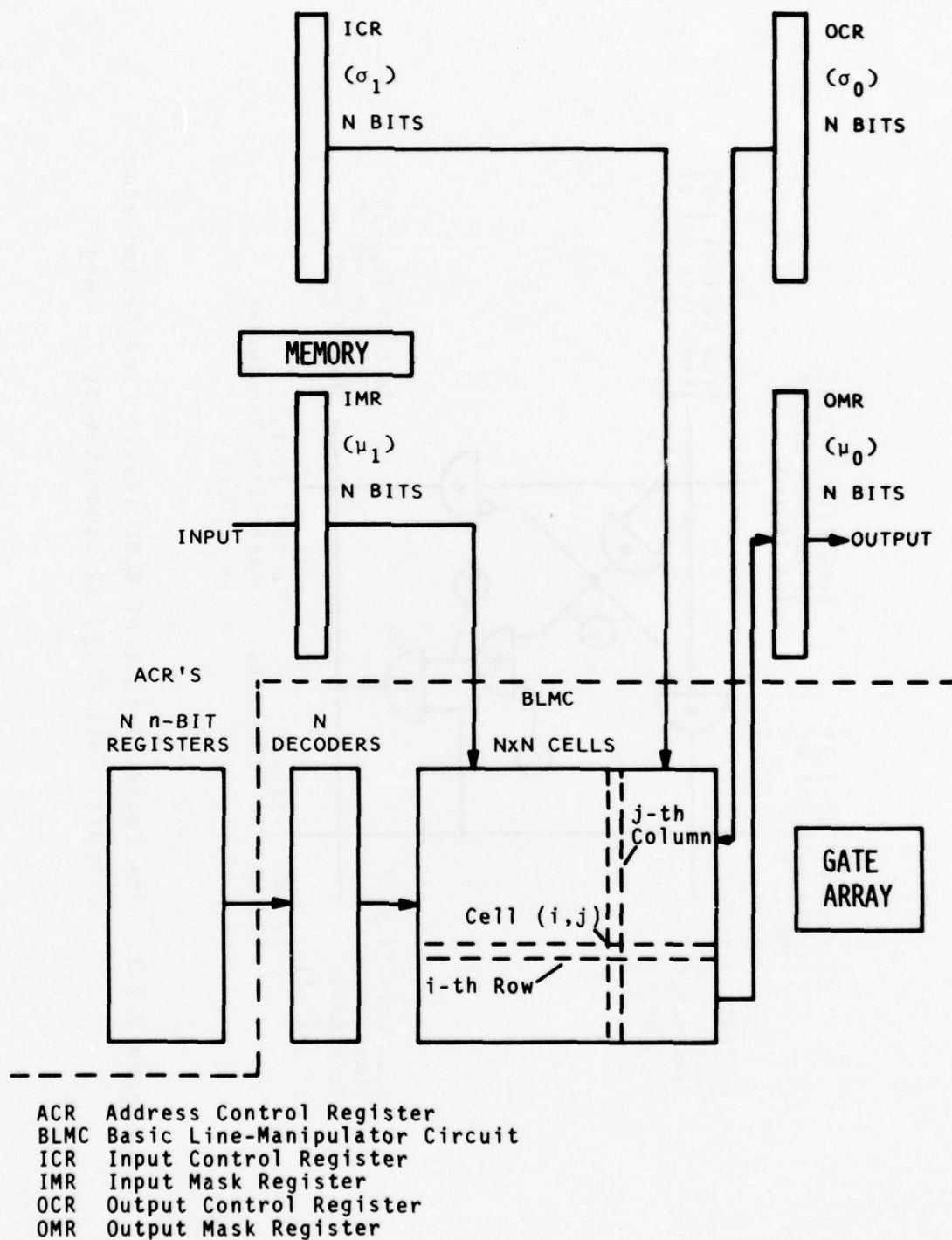


Figure 3.2-1. A versatile line manipulator (VLM) organization suggested by T. Feng.

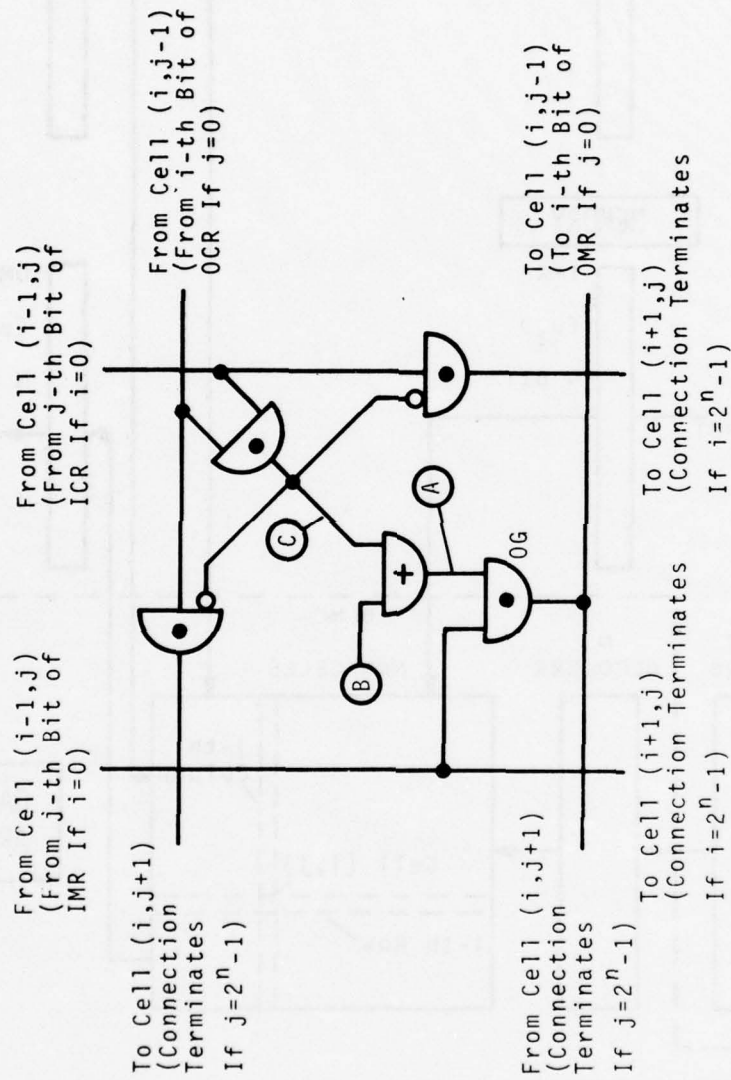


Figure 3.2-2. The logic circuit of BLMC (Basic Line Manipulator Circuit) Cell (i,j) as suggested by T. Feng.

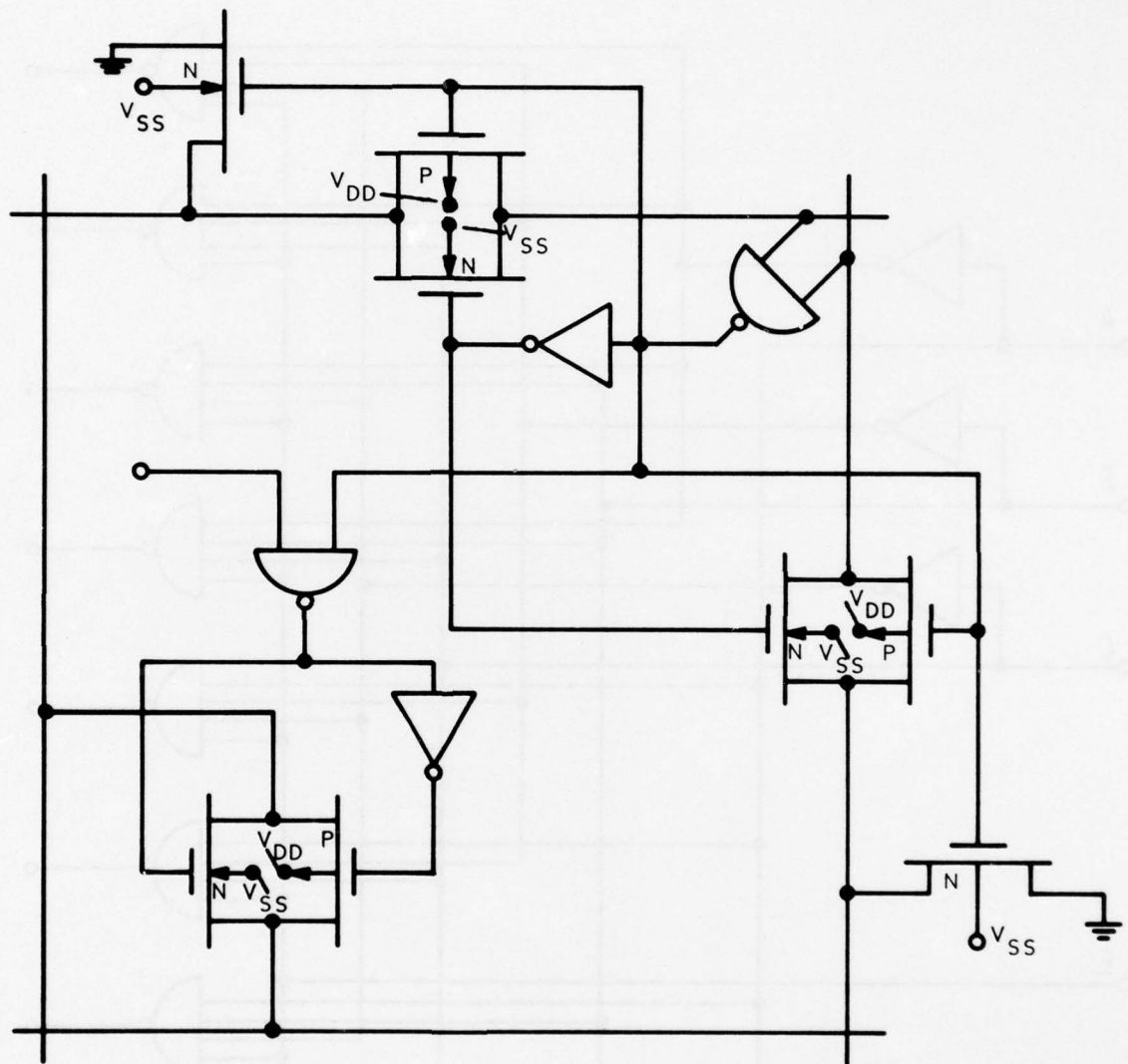


Figure 3.2-3. CMOS design of BLMC cell requiring 10 transistor pairs.



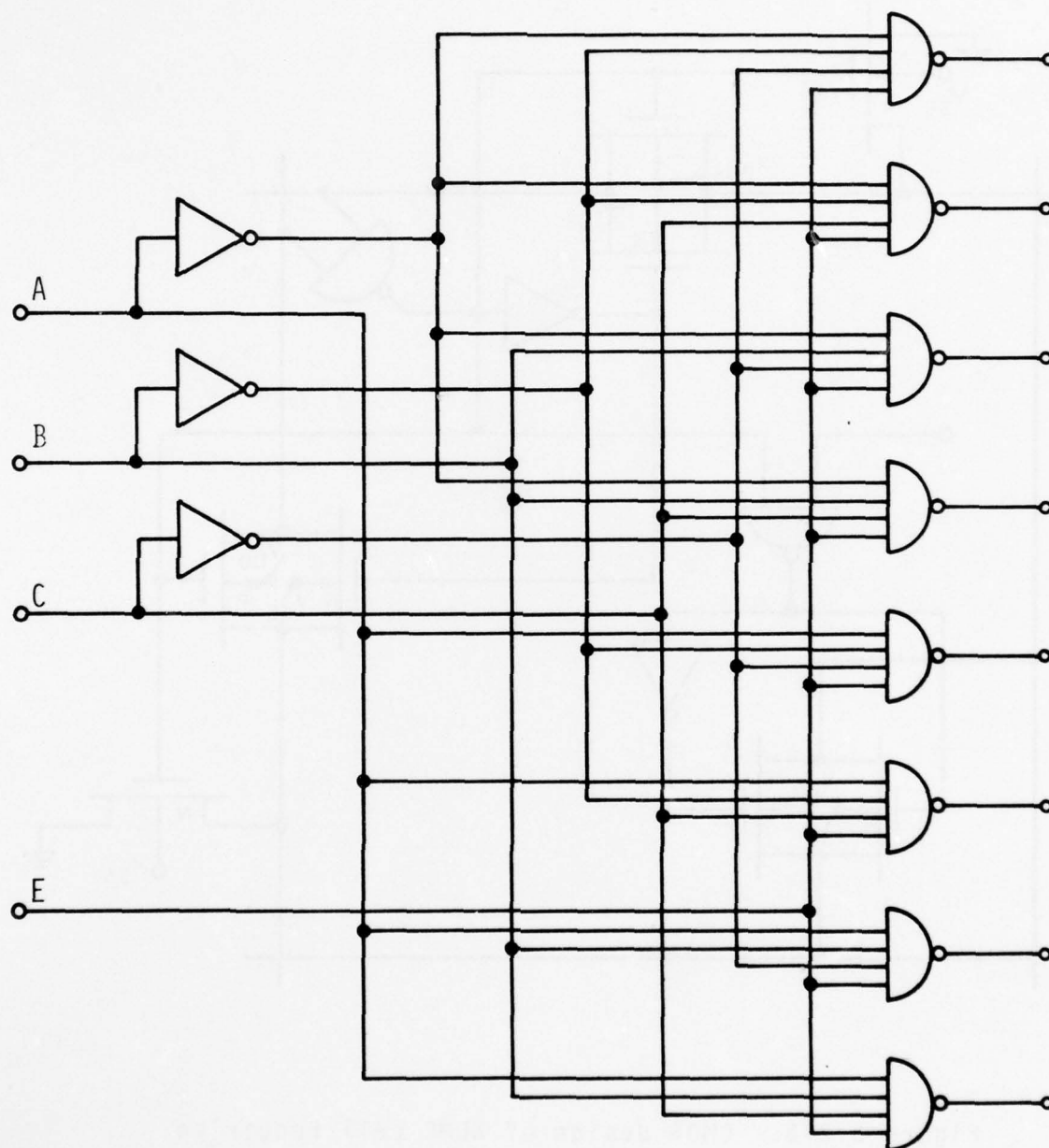


Figure 3.2-4. CMOS design for 3-to-8 decoder with chip enable requiring 35 transistor pairs.

Number Required	Gate Type
$N^2$	AND (output) gate, crossbar switch
$N^2$	OR gate
$3N^2$	AND gate
$N^2$	INVERTER
$N$	$\log_2 N$ -to- $N$ decoder

No significant logic simplification can be effected if the full functional capabilities of the Report by Tse-yun Feng are to be achieved. If the ICR/OCR capability were eliminated, the gate count would be reduced by  $5N^2$  ( $=327,680$  for  $N=256$ ).

The array design work would deal largely with the optimum grouping of these functions to achieve

- (a) lowest-cost producibility,
- (b) easy expansion from  $32 \times 32$  lines to  $256 \times 256$  lines.

The options, oriented towards LSI technology, are as follows:

Option I - Implement an  $N \times N$  array of the gate design as shown in Figure 3.2-2 plus  $N$  ACR decoders on the same chip, where  $N$  lies between 4 and 32.

Option II - Same as Option I, except that the ACR decoders are off the chip and connected to pin B in the drawing. The  $N$  ACR decoders could be on a separate chip.

Option III - Same as Option II, except that the ICR/OCR decoder gates are off the crossbar chip and connected at pin C. These gates could be manufactured on their own LSI chip.

Option IV - Same as Option III, except that the OR gate is also off the chip and its output is connected at pin A.

Exactly which of these options is chosen depends largely on the production-cost trade-off analysis.

Among the possibilities to be considered are not only the  $N \times N$  (square) arrays but also  $N \times M$  arrays, and especially the  $N \times 1$  array, where e.g. one of 32 input lines is connected to a single output line.

To allow the expansion from a 32x32 line array to a 256x256 line array the scheme shown in Figure 3.2-5 may be used. Each 32x32 array would constitute a Subarray (I,J) which contains all its internal decoding plus one "SUBARRAY-SELECT" pin which activates the entire subarray under the control of the overall ARRAY-TO-SUBARRAY decoder, using the 3 most significant bits of the 8-bit address code.

It is technologically feasible to mount a 32x32 line DM gate array on a single PC board, which would then simply be duplicated 64 times to produce the 256x256 line array.

The Memory section of the Data Manipulator requires storage of 14N bits (8 bits of ACR for a 256x256 array, plus 1 bit each for ICR, OCR, IMR, OMR and input and output registers). Obviously, such a small number of circuits does not warrant any custom chip development and the memory would therefore always be implemented with off-the-shelf circuitry, typically bipolar to maintain maximum speed capability for loading, presetting and clearing.

If the decision were made to implement the Gate Array itself in LSI technology, most likely CMOS, using the circuit designs of Figure 3.2-3 and -4 above, it is obvious that more than one BLMC cell should be housed in a single package. The design of Figure 3.2-3 requires approximately 300 sq. mils of circuit area and a 3-to-8 decoder with chip enable according to Figure 3.2-4 would require approximately 700 sq. mils.

A typical LSI chip which could be built immediately would be a 3x8 array, with 24 DM gate cells and three 3-to-8 decoders, using 345 CMOS transistor pairs, with a chip area of less than 100 mils square. The pin assignments would be as follows:

<u>Function</u>	<u>No. of Pins</u>
IMR (data in)	8
OMR (data out)	3
ICR	8
OCR	3
Decoder inputs	9
Chip enables	3
B +	1
Ground	1
Total	36

1, 1	1, 2	1, 3					1, 8
2, 1							
3, 1							
4, 1							
8, 1							8, 8

Figure 3.2-5. Expansion from 32x32  
to 256x256 line array.



This chip could be housed in a popular 40-pin package with a PC board area of under 2.5 sq. in. Very high yield and good reliability could be expected from such a chip and package combination. A 32x32 line gate array would require 44 packages on a PC board of 11x11" (or 6.5x17") with approximately 320 edge-connector pins.

Current state-of-the-art allows 3 to 4 times the above complexity per chip, but the yield might be significantly lower, and more expensive packages (up to 100 pins) would be needed.

It is obvious that a custom LSI implementation of the Data-Manipulator Gate Array is entirely feasible.

The second major design alternative, shown in Figure 3.2-6 is based on the use of an off-the-shelf bipolar 16-to-1 line-select circuit with on-chip address decoding. Using only 4 of these integrated circuits, any of 64 input lines can be selected as the source for a particular output line.

The other major difference between the first and second designs is that in case 2 the ICR/OCR decoding function is removed from the chip and implemented by a special circuit which scans the ICR and OCR registers and converts their contents into the proper ACR settings to produce the same results (the design of the circuit is described later). Since the original ICR/OCR decoding function requires between  $4N^2$  and  $5N^2$  gates, this design modification results in a very substantial savings in logic gates.

This simplicity of the second design was one of the main reasons why it was selected as the preferred approach.

Other reasons were as follows:

If only a relatively small number of circuits is required, off-the-shelf integrated circuits are usually less expensive than custom LSI circuits.

The probability for success of the entire project is higher if no custom LSI circuits are required since the latter are always subject to the risk of a design error or technological production problems.

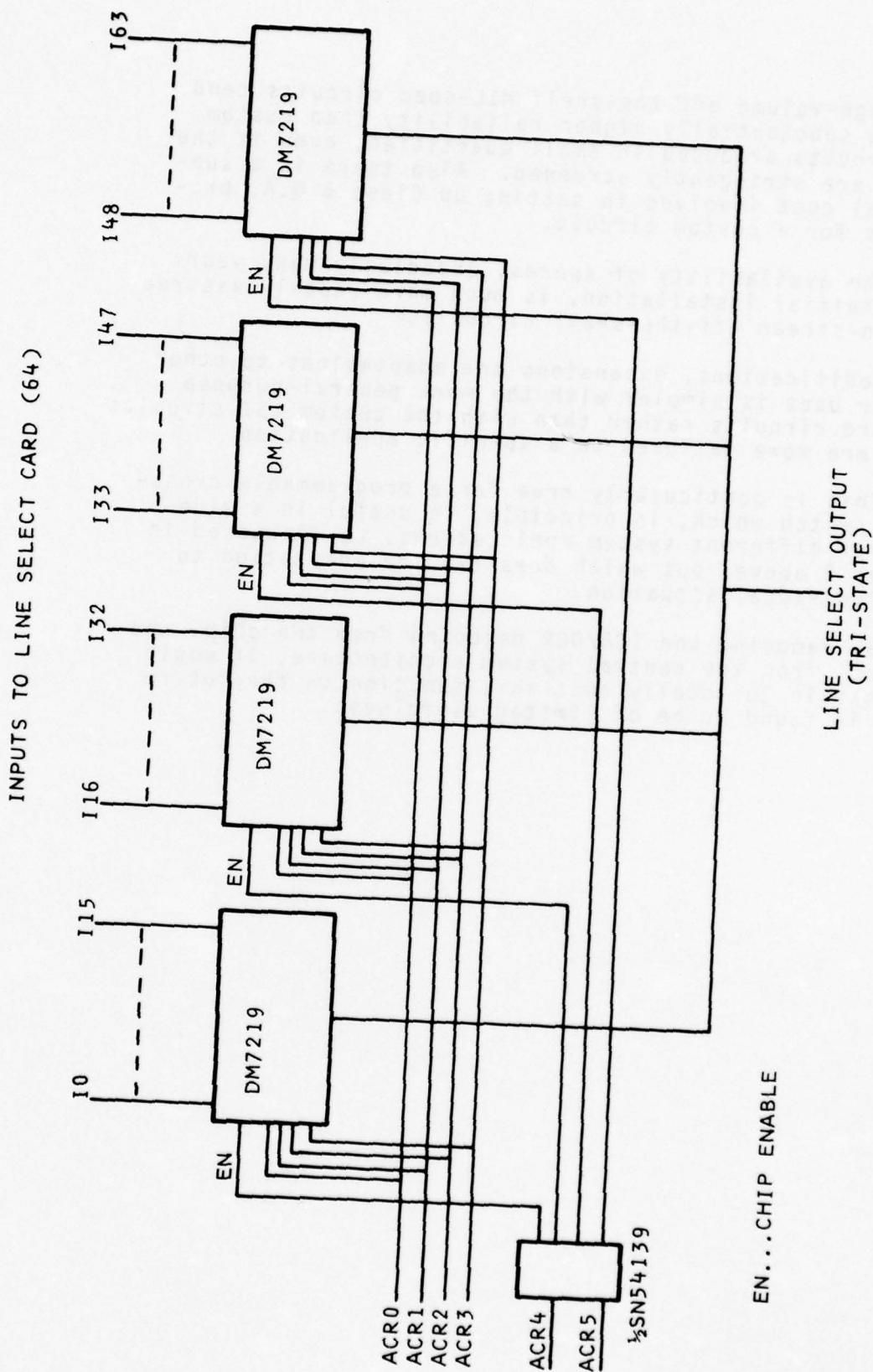


Figure 3.2-6. Implementation of DM function by use of tri-state 16-to-1 multiplexer ICs. Any of 64 input lines can be selected as the source for each output line, using only 4  $\frac{1}{2}$  chips.

High-volume off-the-shelf MIL-spec circuits tend to have substantially higher reliability than custom LSI circuits produced in small quantities, even if the latter are stringently screened. Also there is a substantial cost involved in setting up Class B Q.A. procedures for a custom circuit.

The availability of spares, especially many years after initial installation, is much more readily assured by main-stream off-the-shelf circuits.

Modifications, expansions and adaptations to other similar uses is simpler with the more general-purpose standard circuits rather than with the custom LSI circuits which are more tailored to a specific application.

This is particularly true for a programmable cross-point switch which, in principle, is useful in a wide range of different system applications, as discussed in Chapter 2 above, but which does require adaptation to each individual situation.

By removing the ICR/OCR decoding from the chip, and in fact, from the central system architecture, it would be possible to totally omit this function in the future if it is found to be of limited usefulness.



### 3.3 Description of System Design

A block diagram of the system as it was ultimately implemented is shown in Figure 3.3-1, including the interfaces to the STARAN computer and the self-test circuitry. The Data Manipulator operates under the PIO Control of the STARAN Computer via the PIO Control Interface. The design developed by W. W. Gaertner Research, Inc. is fully ECL and TTL-compatible, so that the Data Manipulator could also be interfaced to other computer systems. The contents of the Input and Output Mask Registers (IMR, OMR), of the Address Control Registers (ACR) and of the Input and Output Control Registers (ICR, OCR), as well as the data to be manipulated, are entered into the Data Manipulator via the 256-bit wide PIO Buffer Interface, of which 64 bits have actually been implemented in the initial equipment. The manipulated data leave the Data Manipulator via the same interface. The instruction repertoire of the Data Manipulator, to load the various address registers and masks, and to start and stop data manipulation and self test are discussed in Chapter 3.5 below.

The Self-Test Computer, a PDP-11/03 with 8K of memory, in conjunction with the Control Module and the Test Data Register (TDR) is capable of loading and reading all registers, including the input and output data registers, and furthermore has been designed to simulate all STARAN instructions, so that it can operate the Data Manipulator in all modes, even if it is not connected to the STARAN computer. This feature makes the Data Manipulator also compatible with the PDP-11 family of minicomputers.

In typical operation, the Address Control Registers (ACR) and the Input and Output Mask Registers (IMR, OMR) would be loaded first, and any parallel data passed to the Data Manipulator thereafter would be manipulated according to the settings of the Address Control Register and would be masked according to the data loaded into the Input and Output Mask Registers. In an alternate mode of operation the ACRs are not loaded, but parallel data are fed to the Input Control Register and Output Control Register (ICR, OCR) and a special instruction from the STARAN PIO Control Interface directs the ICR/OCR-to-ACR Conversion Logic Circuitry to rapidly convert the ICR and OCR settings to appropriate ACR addresses. This step takes less than 4 microseconds in a 64x64-line Data Manipulator. Subsequent to this conversion, the Data Manipulator operates as if the ACRs had been loaded directly.



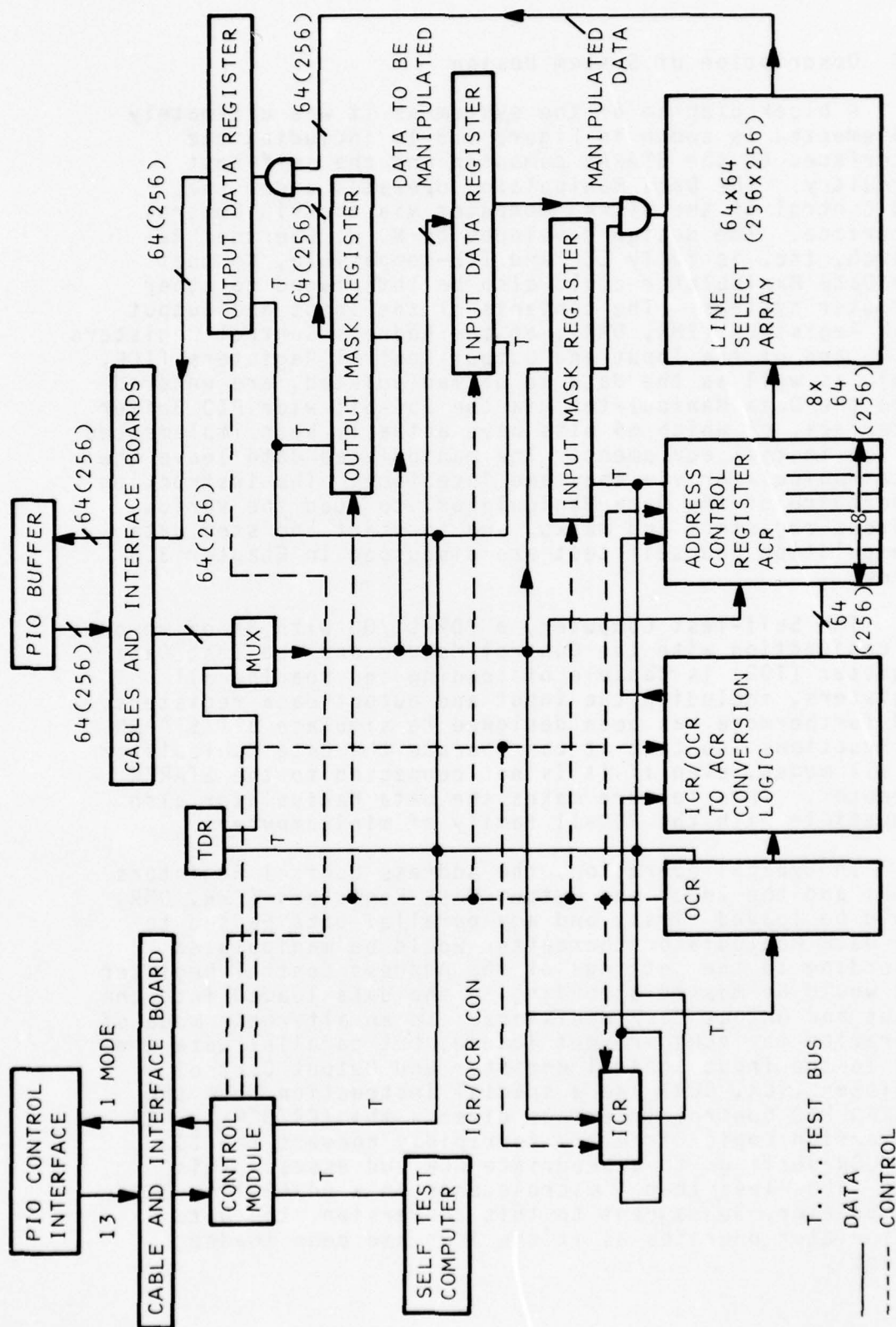


Figure 3.3-1. Block Diagram of Data Manipulator

The delay through the Data Manipulator in the actual manipulation mode is less than 250 ns, i.e. less than one-third of the original specification.

While only a 64x64-line capability was implemented in the initial system, all the design was laid out to allow ready expansion to a full 256x256 system. Specifically, all ACRs were provided with full 8-bit rather than 6-bit capability and the fan-out of all circuits involved in potential future expansion was dimensioned to handle a full system. Thus, expansion can be accomplished strictly by adding more cards of the existing design, rather than requiring new card layouts and changes in circuit design or component choice to provide adequate fan-out.

The instruction repertoire for the Data Manipulator is discussed in detail in Chapter 3.5 below. The basic approach chosen was to select certain STARAN PIO Control instructions and have the Data Manipulator interpret them differently than if they had been sent to another Associative Array. Among the many possibilities the Array Address field in the PIO Control word was chosen since there seemed the least likelihood that an identical instruction might be issued by the STARAN Computer for another purpose and misinterpreted by the Data Manipulator. All indications with actual operation to date indicate that this choice was correct. Furthermore, the Control Module was implemented by wire-wrap boards to allow circuit changes which might have been necessary if one wanted to use STARAN PIO Control instructions different from those originally chosen. The delivered equipment retains this flexibility for any future adaptations which might become desirable as the RADC Parallel/Associative Computer research facility is expanded.

The use of a Self-Test Computer proved an invaluable tool during development, installation, acceptance testing and maintenance of the Data Manipulator. Chapter 5 below discusses the capabilities and use of the Self-Test Program in considerable detail and thereby also provides additional insight into the operation of the Data Manipulator. It made it possible to run the Data Manipulator for many hours in a life-test/burn-in mode at a rate of over 2 million verified manipulations per hour, selected carefully to exercise systematically every circuit in the Data Manipulator, and provide a complete log of all permanent and transient errors. Considering the success of this approach

with the Data Manipulator, and the continuously decreasing cost of mini/microcomputers, it appears that the incorporation of a built-in self-test computer into a major peripheral of a large computer system can substantially reduce the development cost and minimize subsequent problems in integration and maintenance. It also greatly reduces the time which must be made available on the main computer system for the installation and integration of the new add-on module.

Choosing a member of the PDP-11 family of minicomputers as the Self-Test Computer also happens to make the Data Manipulator compatible with this popular type of minicomputer. Or, if the Data Manipulator were to serve as a cross-point switch in a communication system, the Self-Test Computer could perform the function of the switch-setting control computer. Instructions to the control computer could be sent via its standard serial and parallel interfaces. Both Decscope and teletype terminals have been used by W. W. Gaertner Research, Inc. for this purpose.

The details of the electronic and mechanical designs, of the software, physical construction, self-test and maintenance of the Data Manipulator are described in the following paragraphs.

### 3.4 Hardware Design and Operation

#### 3.4.1 Functional System Partitioning

The system is logically and physically divided into the following five sections:

- (i) The STARAN Interface. This circuitry consists of an Input Interface Card (expandable to 4), an Output Interface Card (expandable to 4) and a Control Interface Card.
- (ii) The Line Select Array. This section provides the crossbar-like connection matrix between up to 256 inputs and outputs. It consists of a set of Line Select Cards, each of which interconnects 64 inputs to 16 outputs. The present system uses 4 Line-Select Cards to implement a 64x64 system. An expanded 256x256 system requires 64 of these cards.
- (iii) The Register Section. All of the registers shown in Figure 3.3-1 are physically located on the Register Cards, as well as most of the ICR/OCR to ACR conversion logic. Each Register Card contains an 8-bit slice of all the registers, the associated self-test circuitry, and ICR/OCR-to-ACR conversion circuits. The current 64x64 system has 8 Register Cards. A full-sized 256x256 system would have 32.
- (iv) The Control Module. This module consists of two wire-wrapped cards: The Control Card, and the LSI-11 Interface Card. The latter is physically located inside the PDP-11/03 chassis and the two cards are connected by a 50-conductor ribbon cable.
- (v) The Self-Test Computer. This is a standard DEC PDP-11/03 computer, consisting of an LSI-11 CPU card, an MMV11-A 4K core memory card and a DLV-11 Serial interface card.

#### 3.4.2 STARAN Interface

As mentioned above, the STARAN Interface consists of an Input Interface Card, an Output Interface Card and a Control Interface Card.

The Input and Output Interface cards each provide 64-bit paths for the data and perform the conversion between the differential ECL-compatible signals at the STARAN end and the TTL-compatible signals utilized in the rest of the Data Manipulator. This is shown in detail in Drawings no. 4051-011676-D and 4051-012076-D, "Data Manipulator - Engineering Drawings".



The Control Interface Card (Drawing no. 4051-011576-D) handles a total of 13 signals, which are utilized to control the operation of the Data Manipulator under the direction of a STARAN PIO Control program. 8 of these come from bits 8 to 15 of the PIO control instruction (the Array Address Field); they are interpreted as instructions by the Data Manipulator (SI8L to SI15L). Timing is provided by 4 clocks from the STARAN - WRTMEM, LOAD MASK, LOADX and LOADY. These clocks are stretched on this card to generate TTL pulses approximately 100 nsec wide. The STARAN PIO Control also provides the ARRAY SEL H signal which is used to enable interpretation of STARAN instructions by the Data Manipulator.

In the other direction, the Data Manipulator provides a DM RDY H signal to indicate it is in a mode where it will manipulate input data. This, after conversion to differential ECL-compatible signals, becomes the SUMOR signal for the STARAN PIO control.

#### 3.4.3 Line Select Cards

Each Line Select Card contains 16 multiplexers, each of which can select one of 64 common inputs or can be disabled as shown previously in Figure 3.2-6. The multiplexer outputs are tri-state, so that its output can be connected to the outputs from three other Line Select Cards to form a 256-bit input multiplexer in a full-sized 256x256 system. The detailed logic of the Line Select Card is shown in Drawing no. 4051-070876-D. Page 1 shows the receivers and line drivers used on the 64 input data lines. The 7837 receivers provide high input impedance and input hysteresis to improve the noise immunity. The 54367's provide the fanout needed to drive the 16 multiplexers. Pages 2 through 9 of the drawing each shows two of the 16 64-input multiplexers. Each multiplexer is addressed by the corresponding ACR inputs. The 6 low-order bits address the multiplexer while the 2 high-order bits are used to select the multiplexer if they specify the card address as set by the two jumpers JA and JB (shown at the right end of page 1). Note that for the present 64x64 system, these jumpers are absent on all 4 Line Select Cards corresponding to 0's in bits 6 and 7 of the ACRs.

#### 3.4.4 Register Cards

A block diagram of the register card is given in Figure 3.4.4-1 (Drawing no. 4052-051076-D). 8-bit slices of all the registers are shown and the decoding circuits for the STARAN instructions are contained in the lower left section of the diagram. All of the registers can be read by the Self-Test Computer through multiplexers in the upper right section. In addition, the registers can be loaded by the Self-Test Computer using the decoder and



test data register in the left section to simulate STARAN instructions. The ICR/OCR to ACR conversion circuitry is shown in the lower right section of the drawing; a description of its operation is given in Section 3.4.5 below.

The Test Data Register (TDR) is utilized in Test Mode to simulate the loading of other registers via STARAN PIO instructions.

When a register is to be loaded under STARAN control, the control card provides a REG LDH pulse, and a register address on the INST 0-4 lines, which correspond to 4 of the 8 instruction bits from the STARAN. The data normally comes in on the 8 STARAN DATA IN lines from the STARAN interface card. During Self-Test, when a STARAN instruction is to be simulated, the data from the TDR is used instead.

The ACR's on a register card are implemented as an 8x8 array of D flip flops (54S74). Either a row or a column can be loaded at a time, while all the 64 bits are available in parallel on the backplane to control the Line Select Array. During a load (or clear or preset) ACR instruction from the STARAN, a column of the ACR is loaded on all the register cards corresponding to a bit slice. During the ICR/OCR to ACR conversion process, however, a row is loaded at a time on any one register card. The Self-Test Computer also accesses the ACR's by row.

The Self-Test Computer can load any register on the card. The data enters the register cards on the 8-bit TEST OUT BUS, and the register address on the 4-bit TST AD BUS. Loading takes place at the trailing edge of the TST LD pulse. The outputs of all the registers are combined via tri-state buffers and multiplexers to produce the tri-state 8-bit TEST IN BUS, to allow the self-test computer to read any register.

#### 3.4.5 ICR/OCR to ACR Conversion

A simplified block diagram of the ICR/OCR to ACR conversion system is shown in Figure 3.4.5-1. This system was chosen over several alternatives, primarily because the timing is less critical.

ICR/OCR to ACR conversion begins by transferring the contents of the OCR to the OCR BUFFER. This is done to avoid losing the information in the OCR during the conversion. The conversion consists of matching each 1 in the ICR with a 1 in the OCR and then loading the binary value of the ICR 1's position into the ACR specified by the OCR 1's position.

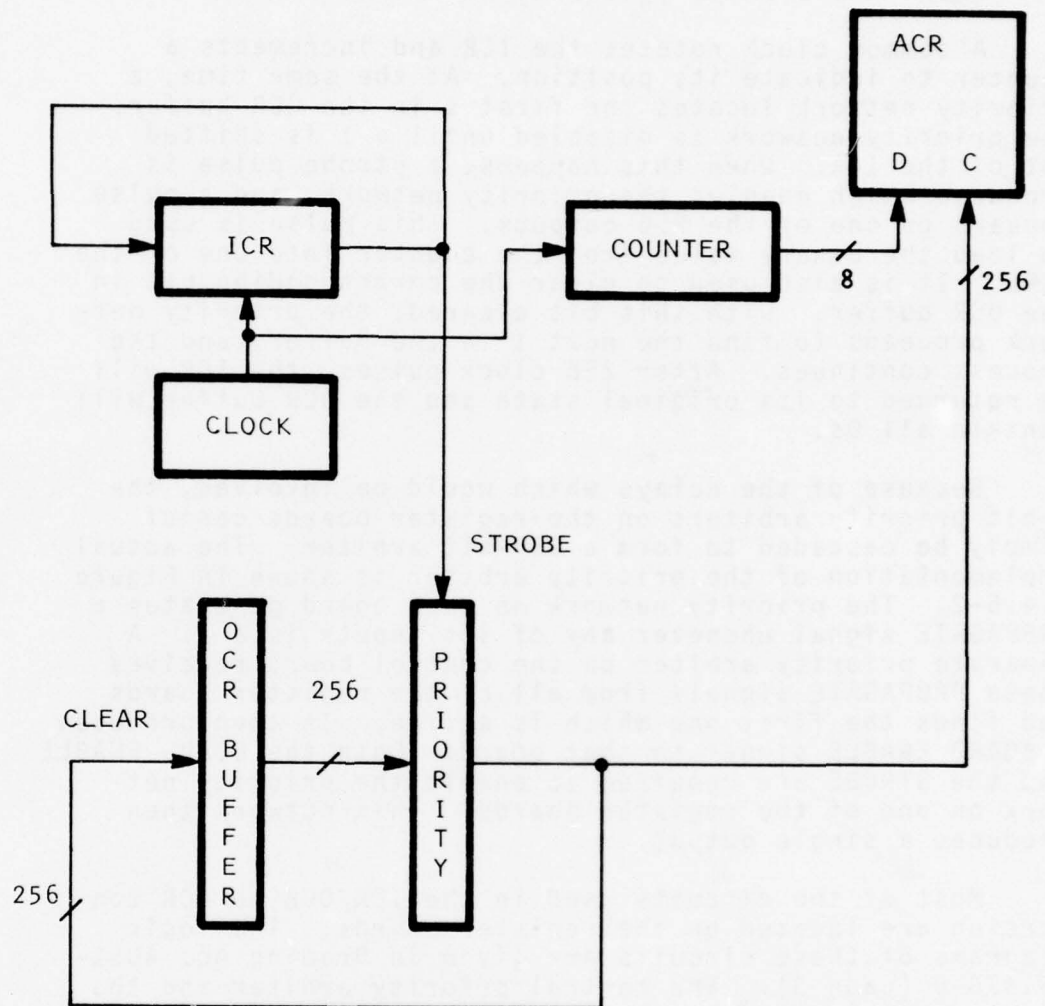


Figure 3.4.5-1. ICR/OCR to ACR Conversion



The conversion is accomplished as follows:

A common clock rotates the ICR and increments a counter to indicate its position. At the same time, a priority network locates the first 1 in the OCR buffer. The priority network is disabled until a 1 is shifted out of the ICR. When this happens, a strobe pulse is produced which enables the priority network, and a pulse appears on one of the 256 outputs. This pulse is used to load the binary value from the counter into one of the ACRs. It is also used to clear the corresponding bit in the OCR buffer. With this bit cleared, the priority network proceeds to find the next 1 in the buffer, and the process continues. After 256 clock pulses, the ICR will be returned to its original state and the OCR buffer will contain all 0s.

Because of the delays which would be involved, the 8-bit priority arbiters on the register boards cannot simply be cascaded to form a 256-bit arbiter. The actual implementation of the priority arbiter is shown in Figure 3.4.5-2. The priority network on each board generates a PROPAGATE signal whenever any of its inputs is a 1. A separate priority arbiter on the control board receives these PROPAGATE signals from all of the register boards and finds the first one which is active. It then produces a BOARD ENABLE signal to that board. Both the BOARD ENABLE and the STROBE are required to enable the priority network on one of the register boards. This network then produces a single output.

Most of the circuits used in the ICR/OCR to ACR conversion are located on the register boards. The logic diagrams of these circuits are given in Drawing no. 4051-051476-D (page 3). The central priority arbiter and the counter are located on the Control Board (Drawing no. 4051-052076-D; page 1).

#### 3.4.6 Control Card

This is a wire-wrapped card that forms the nucleus of the Data Manipulator. The functions performed by the Control Card include the following:

- (i) It provides timing and control signals for the register cards.
- (ii) It interprets control instructions from the STARAN.
- (iii) The status (or modes of operation) of the Data Man-

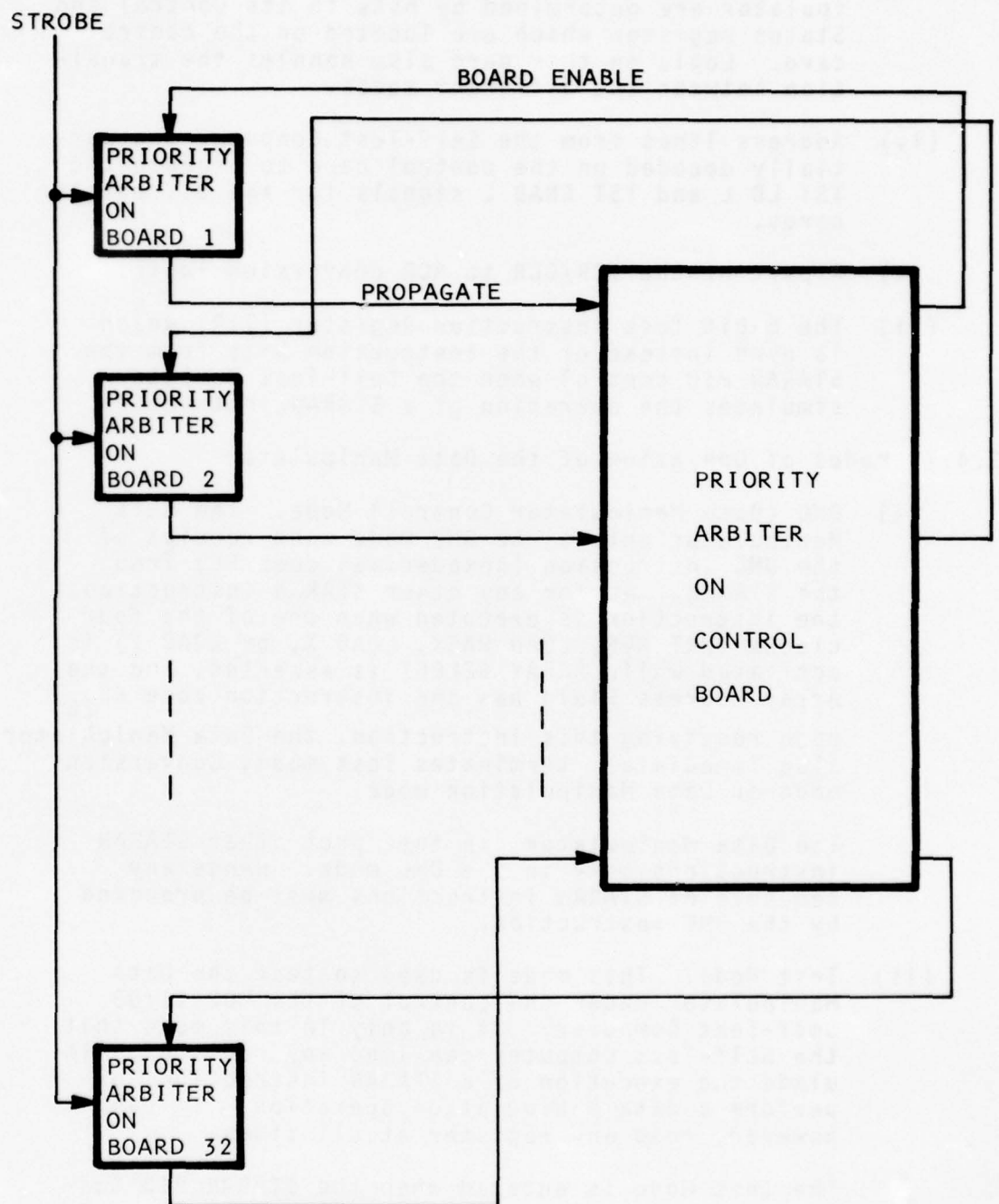


Figure 3.4.5-2. Priority Arbiter

ipulator are determined by bits in its Control and Status register which are located on the control card. Logic on this card also handles the transition between the different modes.

- (iv) Address lines from the Self-Test Computer are partially decoded on the control card to produce the TST LD L and TST ENAB L signals for the 32 register cards.
- (v) A part of the ICR/OCR to ACR conversion logic.
- (vi) The 8-bit Test Instruction Register (TIR) which is used instead of the instruction bits from the STARAN PIO control when the Self-Test Computer simulates the operation of a STARAN instruction.

#### 3.4.7 Modes of Operation of the Data Manipulator

- (i) DMC (Data Manipulator Control) Mode. The Data Manipulator enters the DMC mode upon receipt of the DMC instruction (hexadecimal code FF) from the STARAN. As for any other STARAN instruction, the instruction is executed when one of the four clocks (WRT MEM, LOAD MASK, LOAD X, or LOAD Y) is activated while ARRAY SELECT is asserted, and the Array Address Field has the instruction code FF<sub>16</sub>). Upon receiving this instruction, the Data Manipulator also immediately terminates Test mode, Conversion mode or Data Manipulation mode.

The Data Manipulator can interpret other STARAN instructions only in the DMC mode. Hence any sequence of STARAN instructions must be preceded by the DMC instruction.

- (ii) Test Mode. This mode is used to test the Data Manipulator under the control of the PDP-11/03 Self-Test Computer. It is only in this mode that the Self-Test Computer can load any register, simulate the execution of a STARAN instruction, or perform a data manipulation operation. It can, however, read any register at all times.

The Test Mode is entered when the STARAN PIO Control issues the Start Self Test (code F3<sub>16</sub>) or Resume Self Test (code F5<sub>16</sub>) instruction to the Data Manipulator. The Test Mode can also be set under the control of a program in the Self-Test Computer. This is normally done during the initialization of

the Self-Test program, and the STARAN controls the setting and clearing of the Test Mode thereafter via the instructions  $F3_{16}$ ,  $F5_{16}$  and  $FF_{16}$ .

Setting or clearing Test Mode produces an interrupt on the Self-Test Computer, provided the interrupts are enabled. It also controls the Test Mode light on the front panel.

- (iii) Conversion Mode. The Data Manipulator enters this mode on executing a Perform Conversion instruction (code  $F9_{16}$ ). It stays in this mode for approximately 4 microseconds during which time the ICR/OCR to ACR conversion takes place. This has already been described in Section 3.4.5.
- (iv) Data Manipulation Mode. This is the default mode, present when none of the other modes described above are active. In this mode, the IR is loaded with data from the PIO Buffer whenever a clock is received from the STARAN, and the OR is loaded with the resulting manipulated data after a delay of approximately 250 nsec. The OR can subsequently be read back by the STARAN.
- (v) Off-Line Mode. This mode is controlled by the off-line switch on the front panel. In Off-Line Mode, the instruction and clock lines from the STARAN are disconnected from the Data Manipulator and the Self-Test Computer can simulate any instruction by loading its code into the TIR and setting the SIX bit in the CSR. In the On-Line (not Off-Line) Mode, the Self-Test Computer can still simulate most STARAN instructions in Test Mode, but not those connected with Test Mode itself ( $DMC-FF_{16}$ , Start Self-test -  $F3_{16}$ , Resume Self Test -  $F5_{16}$ ).

#### 3.4.8 LSI-11 Interface Card

This wire-wrapped card is physically located inside the PDP-11/03 computer. It is connected to the Control Card via a 50-conductor ribbon cable. Through this card, and logic elsewhere in the system, the Self-Test Computer has access to all the registers in the Data Manipulator, including some registers such as the TIR and TDR which are used only for self-test. All these registers appear in the address space of the Self-Test Computer, as shown in Drawing no. 4051-070676-D, and can be read or written like memory locations. However, since an 8-bit data path is used in the Data Manipulator, these locations can be accessed properly only by byte



instructions.

The LSI-11 Interface Card contains logic to perform the following functions:

- (i) Recognize addresses relevant to the Data Manipulator and respond to them.
- (ii) Provide an interface between the 16 bit bidirectional bus of the Test Computer and the 8-bit unidirectional Test In and Test Out buses.
- (iii) Latch the relevant addresses, and drive the address lines.
- (iv) Generate timing signals for reading or writing registers on other cards.
- (v) Generate interrupts and implement the interrupt enable logic.
- (vi) Implement part of the CSR logic.

#### 3.4.9 Control and Status Register (CSR)

The self-test computer can monitor and control the operation of the Data Manipulator through the Control and Status Register (CSR: address - 161000).

The different CSR bits and their functions are described next:

- (i) Bit 0 - SIX (Staran Instruction eXecute). Writing this bit causes the Data Manipulator to simulate the execution of the STARAN Instruction whose code is given by the contents of the TIR (address 161002). This is operational only in Test Mode or Off-Line Mode.
- (ii) Bit 1 - Reset. Writing a 1 in this bit resets various flip flops in the Data Manipulator.
- (iii) Bit 2 - CNV CLK (conversion clock). Writing a 1 in this bit generates a clock for the ICR/OCR to ACR conversion system. Used only in Test Mode when CNV SSTP (bit 3) is set.
- (iv) Bit 3 - CNV SSTP (conversion single-step). When this bit is a 1 and the conversion operation is started in Test Mode, the normal conversion clock is replaced with a pseudo-clock produced under program control by the CNV CLK bit (bit 2). This

can be useful for debugging the conversion logic.

- (v) Bit 5 - DMC (Data Manipulator Control). This is a read-only bit that is a '1' when the Data Manipulator is in DMC mode.
- (vi) Bit 6 - Off Line. This is a read-only bit, which is a '1' when the off-line switch on the front panel is activated.
- (vii) Bit 7 - CON (Conversion Mode). This is a read-only bit that is at a '1' when the Data Manipulator is in the Conversion Mode.
- (viii) Bit 8 - DMX (Data Manipulation eXecute). Writing a '1' into this bit in the Test Mode or Off-Line Mode causes a data manipulation operation to be performed, using the TDR as the input.
- (ix) Bit 12 - IE CT (Interrupt Enable - Clear Test Mode). If this bit is a '1', an interrupt is caused when Test Mode is cleared.
- (x) Bit 13 - IE ST (Interrupt Enable - Set Test Mode). If this bit is a '1', an interrupt is generated when Test Mode is set.
- (xi) Bit 14 - TM (Test Mode). Read only bit; is a '1' when the Data Manipulator is in Test Mode.
- (xii) Bit 15 - ST TST (Start Test Mode). This read-only bit is set when the Data Manipulator enters the Test Mode by executing the Start Self Test instruction ( $F3_{16}$ ). This feature is not utilized by the self-test program in the current system.

### 3.5 Data Manipulator Software

The Data Manipulator operates under the direction of the STARAN PIO Control. It examines the Array Address Field (bits 8-15) of the PIO control word and interprets it as an instruction. One of the four PIO clocks, WRT MEM, LOAD MASK, LOAD X or LOAD Y is used for timing. The Data Manipulator looks to the STARAN PIO Control like one of the associative arrays. The corresponding Array-Select signal is utilized to enable the interpretation of instructions.

The instruction set of the Data Manipulator is shown in Figure 3.5-1. To execute any of these instructions other than instruction no. 37 (Enter DMC mode, code FF<sub>16</sub>) the Data Manipulator must already be in the DMC Mode (see Section 3.4.7). Thus any sequence of instructions is preceded by an Enter DMC instruction.

The Data Manipulator leaves the DMC mode upon executing any of the instructions numbered 38, 39, 40, 41 (codes F3<sub>16</sub>, F5<sub>16</sub>, F1<sub>16</sub>, F9<sub>16</sub>).

To perform data manipulations, a PIO Control Program first issues an Enter DMC instruction (no. 37). Subsequently, instructions nos. 9 through 16 may be used to preset the ACR to all "1s", instructions 17 through 24 to clear the ACRs, instructions 26, 27, 29, 30, 32, 33, 35 or 36 to preset or clear the ICR, OCR, IMR or OMR. None of these instructions require any data from the STARAN PIO buffer. This capability is not contained in the original Data Manipulator specifications but it obviously speeds up the switch-setup process by not requiring the loading of all 0s or 1s just to clear or preset a given set of registers. If, as is frequently required, the various registers are to be loaded with specific data, the STARAN computer would first load these data into the PIO Buffer and enable the buffer output. The PIO Control would then use one of instructions 1 through 8 or 25, 28, 31 or 34 to load one of the ACRs or an ICR, OCR, IMR or OMR. Note that any register needs to be loaded only if the new desired settings are different from the ones used during previous manipulations. Specifically, while it takes 8 bits to specify a full source address in an ACR, changing from one of the major Data Manipulation functions discussed in Chapter 2 to another may require the loading of only one or two out of the 8 bits.

STARAN INSTRUCTIONS FOR DATA MANIPULATOR  
(Bits 8-15 of the 32-bit STARAN  
instruction word)

No.	Instruction		Octal	Hex
1	Load	ACR-0	010	08
2	Load	ACR-1	030	18
3	Load	ACR-2	050	28
4	Load	ACR-3	070	38
5	Load	ACR-4	110	48
6	Load	ACR-5	130	58
7	Load	ACR-6	150	68
8	Load	ACR-7	170	78
9	Preset	ACR-0	002	02
10	Preset	ACR-1	022	12
11	Preset	ACR-2	042	22
12	Preset	ACR-3	062	32
13	Preset	ACR-4	102	42
14	Preset	ACR-5	122	52
15	Preset	ACR-6	142	62
16	Preset	ACR-7	162	72
17	Clear	ACR-0	004	04
18	Clear	ACR-1	024	14
19	Clear	ACR-2	044	24
20	Clear	ACR-3	064	34
21	Clear	ACR-4	104	44
22	Clear	ACR-5	124	54
23	Clear	ACR-6	144	64
24	Clear	ACR-7	164	74

Figure 3.5-1



STARAN INSTRUCTIONS FOR DATA MANIPULATOR  
(Bits 8-15 of the 32-bit STARAN  
instruction word)

No.	Instruction	Octal	Hex
25	Load ICR	210	88
26	Preset ICR	202	82
27	Clear ICR	204	84
28	Load OCR	230	98
29	Preset OCR	222	92
30	Clear OCR	224	94
31	Load IMR	250	A8
32	Preset IMR	242	A2
33	Clear IMR	244	A4
34	Load OMR	270	B8
35	Preset OMR	262	B2
36	Clear OMR	264	B4
37	Enter Data Manipulator Control mode (DMC)	377	FF
38	Start Self-Test	363	F3
39	Resume Self-Test	365	F5
40	Enter Data Manipulation Mode (no ICR/OCR conversion)	361	F1
41	Perform ICR/OCR Conversion and Enter Data Manipulation Mode	371	F9

Figure 3.5-1 (continued)

In any specific case, either the ACRs or the ICR and OCR are loaded. If the ACR mode has been chosen, and all ACRs have been loaded, instruction no. 40 (Enter Data Manipulation Mode) is issued by the STARAN PIO Control, and thereafter any data presented by the STARAN computer at its PIO Buffer will be manipulated, i.e. processed through the Input Mask, Line-Select Array and Output Mask every time a clock is received.

If the ICR/OCR mode is chosen and both the ICR and OCR have been loaded, instruction 41 (Perform ICR/OCR-to-ACR Conversion and Enter Data Manipulation Mode) is issued by the STARAN PIO Control and the ICR/OCR-to-ACR Conversion Logic Circuitry rapidly (4 microseconds) converts the ICR and OCR settings to the appropriate ACR settings. Thereafter, any data presented at the PIO Buffer by the STARAN Computer will be manipulated whenever a clock is received.

A sequence of STARAN instructions to write data to an array and read it back can be used to perform data manipulation if the array number corresponds to the logical location of the Data Manipulator. Similarly, a block of data in an associative array can be "manipulated" by a program which exchanges the contents of two arrays if one of the two arrays corresponds to the Data Manipulator. Utilizing the bidirectional transfer capability of the PIO logic this can be accomplished by an inner program loop of three instructions; i.e., a 64 bit word can be manipulated every three STARAN instruction cycles (approximately 1 microsec) with the current system, and a 256 bit word could be processed in the same time by a full-sized Data Manipulator.

#### 4. PHYSICAL CONSTRUCTION

The Data Manipulator is housed in a standard 24 inch rack (external dimensions - 78" high, 27" wide, 30" deep). It is connected to the STARAN PIO control cabinet by seven 50-conductor flat twisted-pair cables. See Figure 4.0-1 for a view of the RADC STARAN facility including the Data Manipulator at the right.

The Data Manipulator cabinet is designed to accommodate a 256-bit system. The current 64-bit system can be upgraded by adding cards and edge-connectors, and wire-wrapping the rest of the back-plane. The cabinet contains space for 126 printed-circuit cards arranged in three bays of 42 slots each. Each card is typically 16" x 20" and plugs into a 216 pin edge-connector. The current 64-bit system utilizes only 18 of the available card positions. Figure 4.0-2 shows the inside of the rack for the 64-bit system; the back-plane is shown in Figure 4.0-3. The configuration of the cards in a full 256-bit Data Manipulator is detailed in Drawing no. 4051-052876-D (Data Manipulator - Engineering Drawings).

The bulk of the Data Manipulator logic is contained on two types of printed circuit cards: the Register Cards and the Line Select Cards. A 256-bit system contains 32 Register Cards and 64 Line Select Cards, while the present 64-bit implementation contains 8 Register Cards and 4 Line Select Cards.

The operation of the Register Card has been described in Section 3.4.4. Physically, it is a 16"x20" double-sided P.C. card with 153 16-pin and 14-pin ICs. The ground and power distribution system was implemented with Rogers laminated bus strips. This helped achieve high packing density in spite of the relatively random nature of the Register Card logic, and also satisfied the noise-suppression requirements of the large number of Schottky TTL ICs on the card. A photograph of a Register Card is shown in Figure 4.0-4. The edge stiffeners provide the rigidity required to insert and extract a pc card of this size. The stiffeners at the back end of the card are made of L-shaped extrusions. Besides providing physical strength, and supporting the card handles used to insert and guide the cards, they also create a barrier to contain the air flow so that it is possible to maintain the air-cooling of the system even when the cabinet door is open, which is a very attractive feature for system debugging and maintenance.

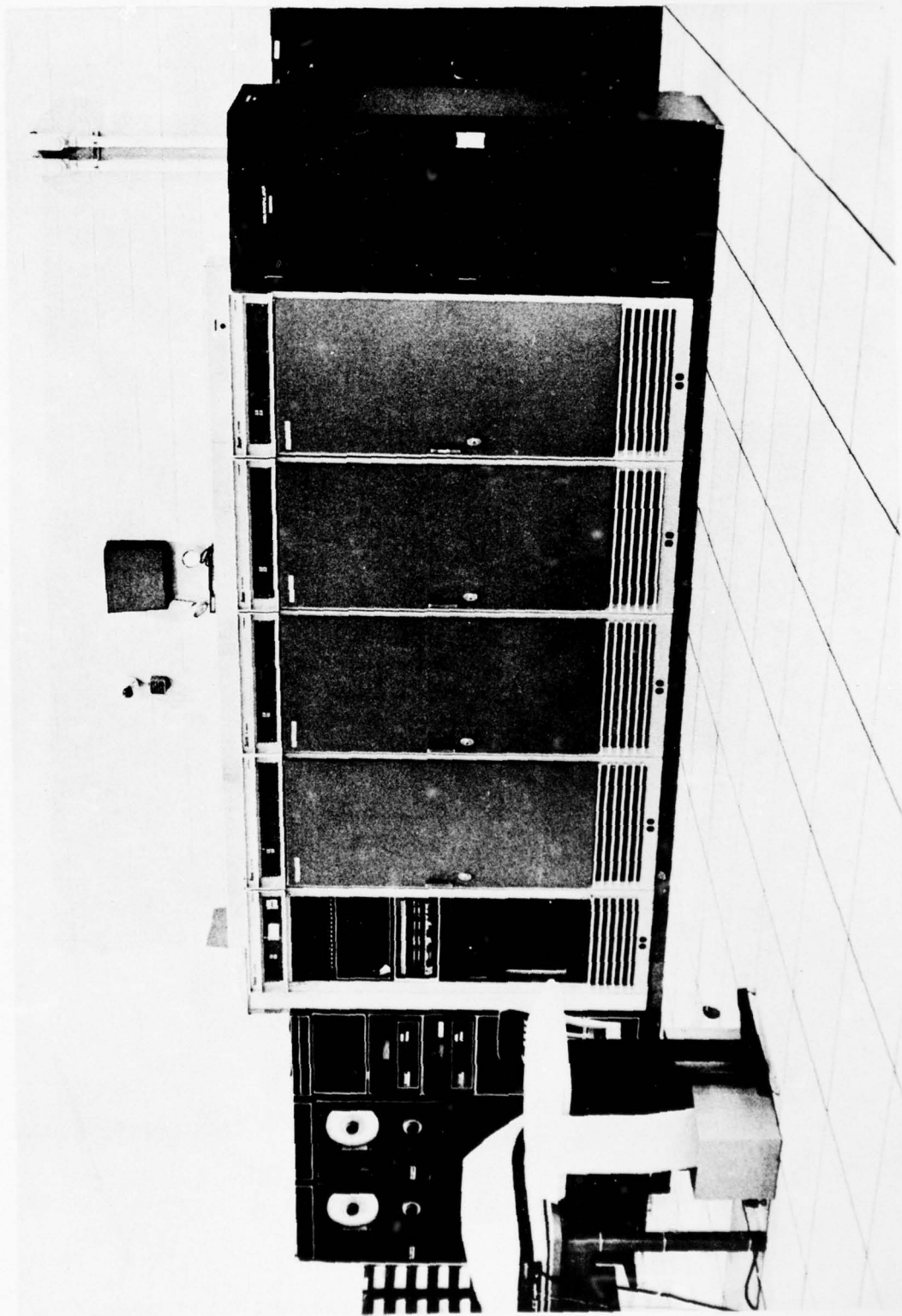


Figure 4.0-1. The RADC STARAN facility with Data Manipulator installed (right-most cabinet)



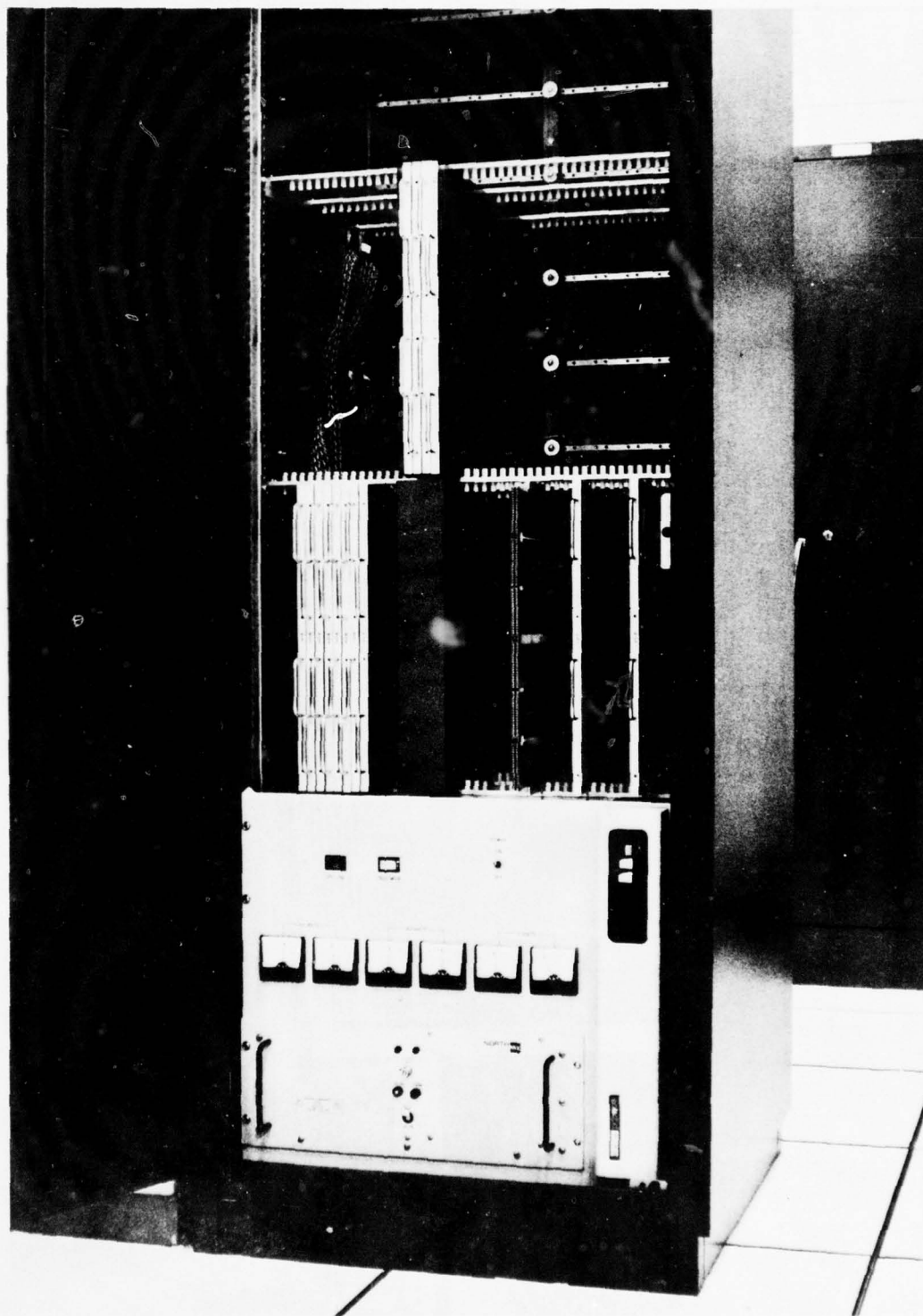


Figure 4.0-2. Data Manipulator with cabinet door open.



Figure 4.0-3. Data Manipulator Back Plane.

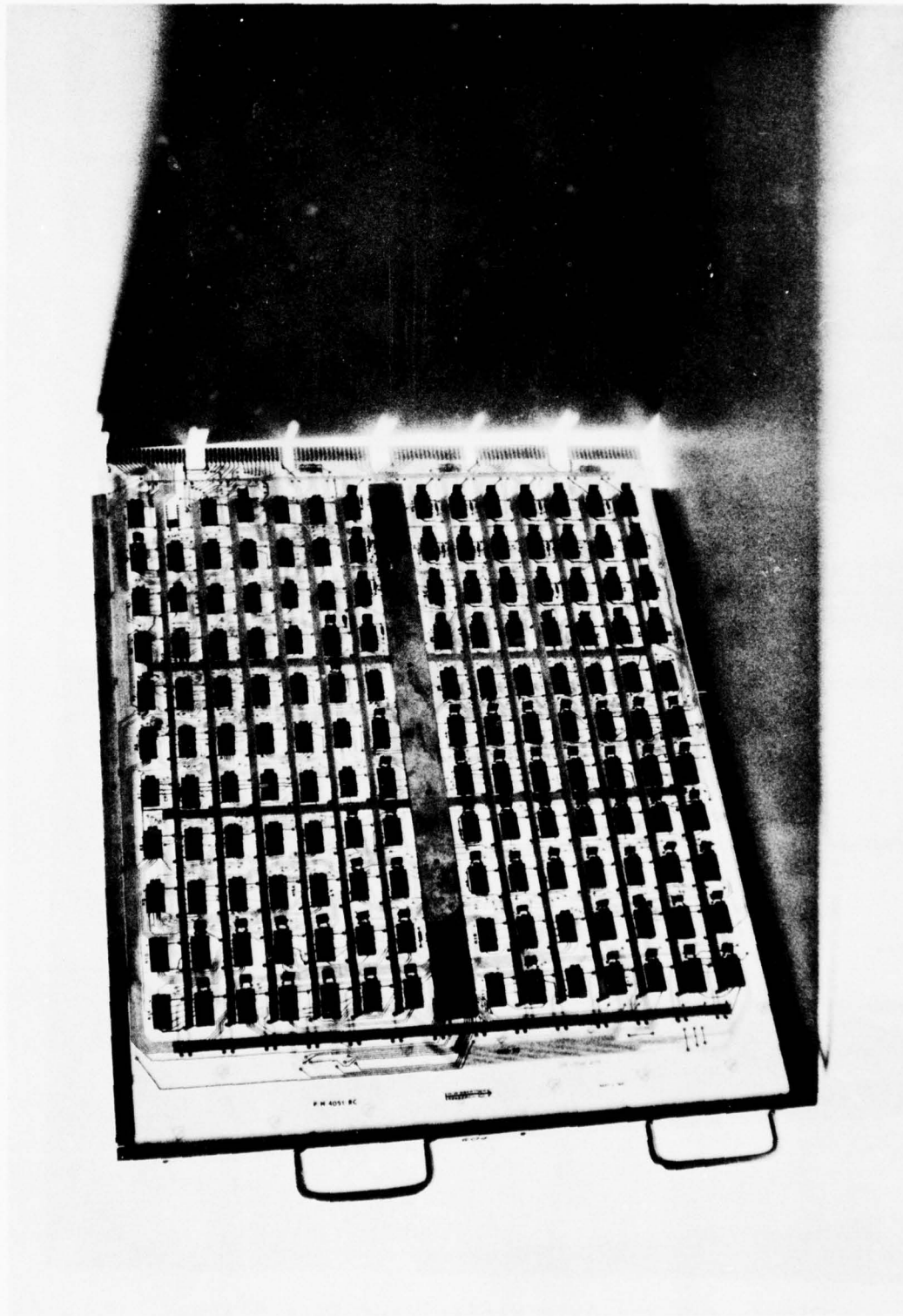


Figure 4.0-4. Register Card

The Line Select Cards implement the basic cross-bar switching function of the Data Manipulator as described in Section 3.4.3. It has been designed as a 16" x 20" double-sided pc board. It holds 64 24-pin ICs, 56 16-pin ICs and 8 14-pin ICs. Since the logic of the Line Select Card is more orderly, and it does not contain any Schottky TTL, it was possible to include the ground and power distribution in the printed circuit pattern. The same kind of L-shaped back-edge stiffeners are used as described above to maintain the air flow when the cabinet door is open. A picture of a Line Select Card is shown in Figure 4.0-5.

Wire-wrap boards were used for the interface cards and the control card to provide flexibility of operation and the ability to adapt the system to other host computers. Photographs of these cards are shown in Figures 4.0-6 through -8. In the case of wire-wrap cards, two slots are taken up because of the height of the wire-wrap pins on the back side of the cards. A 50-conductor ribbon cable connects the control card to an interface card inside the PDP-11/03 test computer. The PDP-11/03 can be seen at the bottom right corner of the rack in Figure 4.0-2.

The bottom of the rack contains the three power supplies (5 volts, -5.2 volts, -2 volts), a blower, and the PDP-11/03 computer. Space for additional blowers and fans is available at the bottom and the top of the cabinet. The front panel contains voltage and current meters for the power supplies, a main switch/circuit breaker and other switches and indicators. These are described in Section 4, Operating Instructions, of the Data Manipulator System Manual.



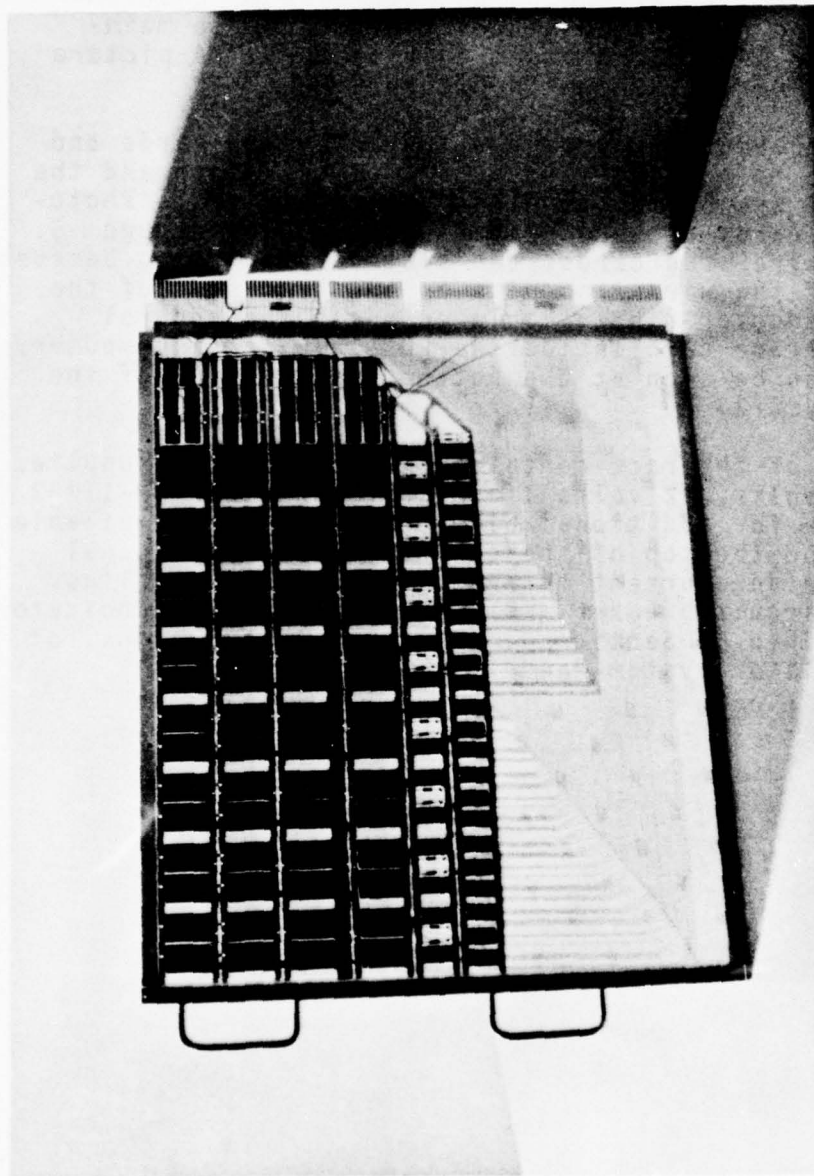


Figure 4.0-5. Line Select Card.

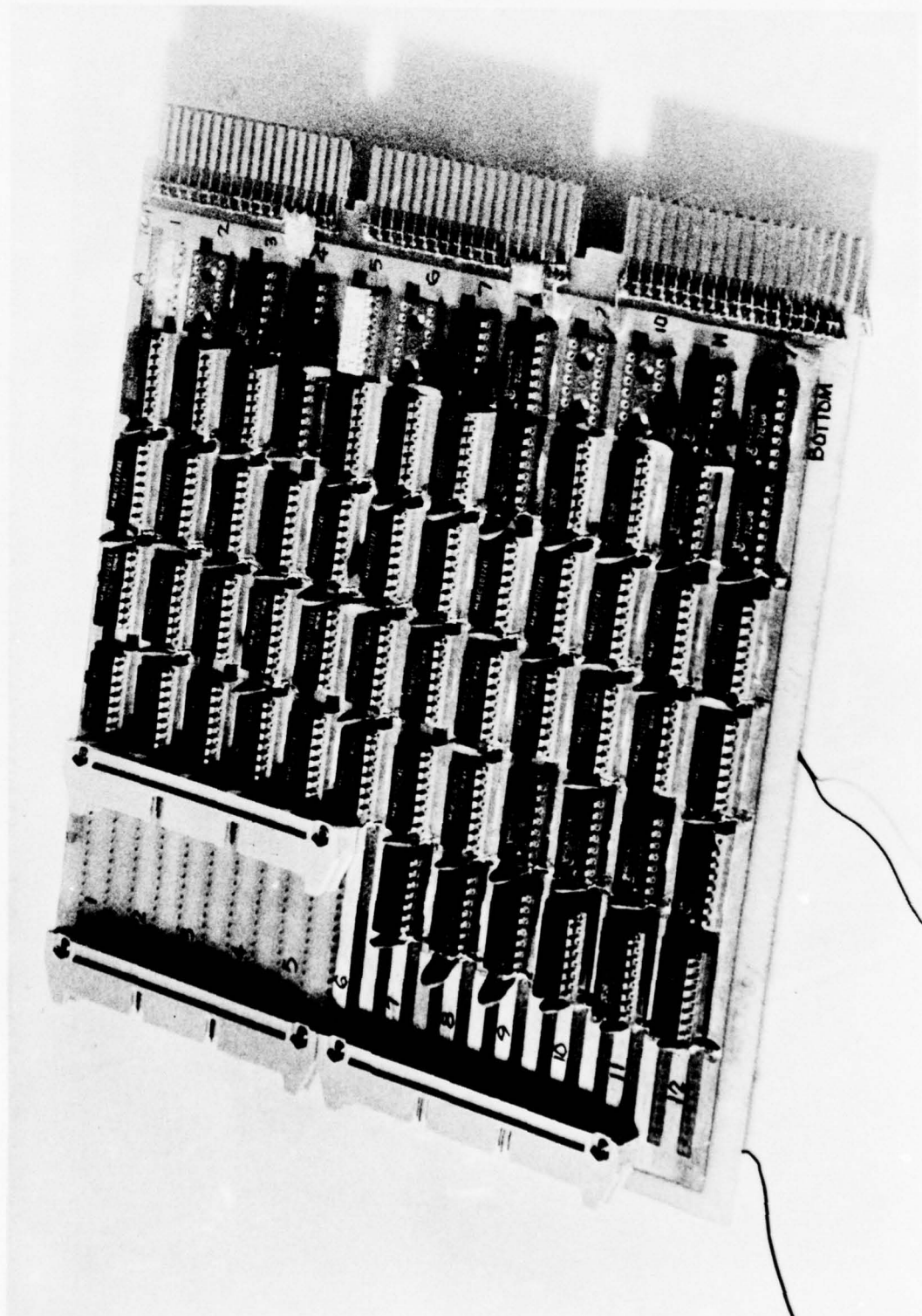


Figure 4.0-6. STARAN Interface Card - Input

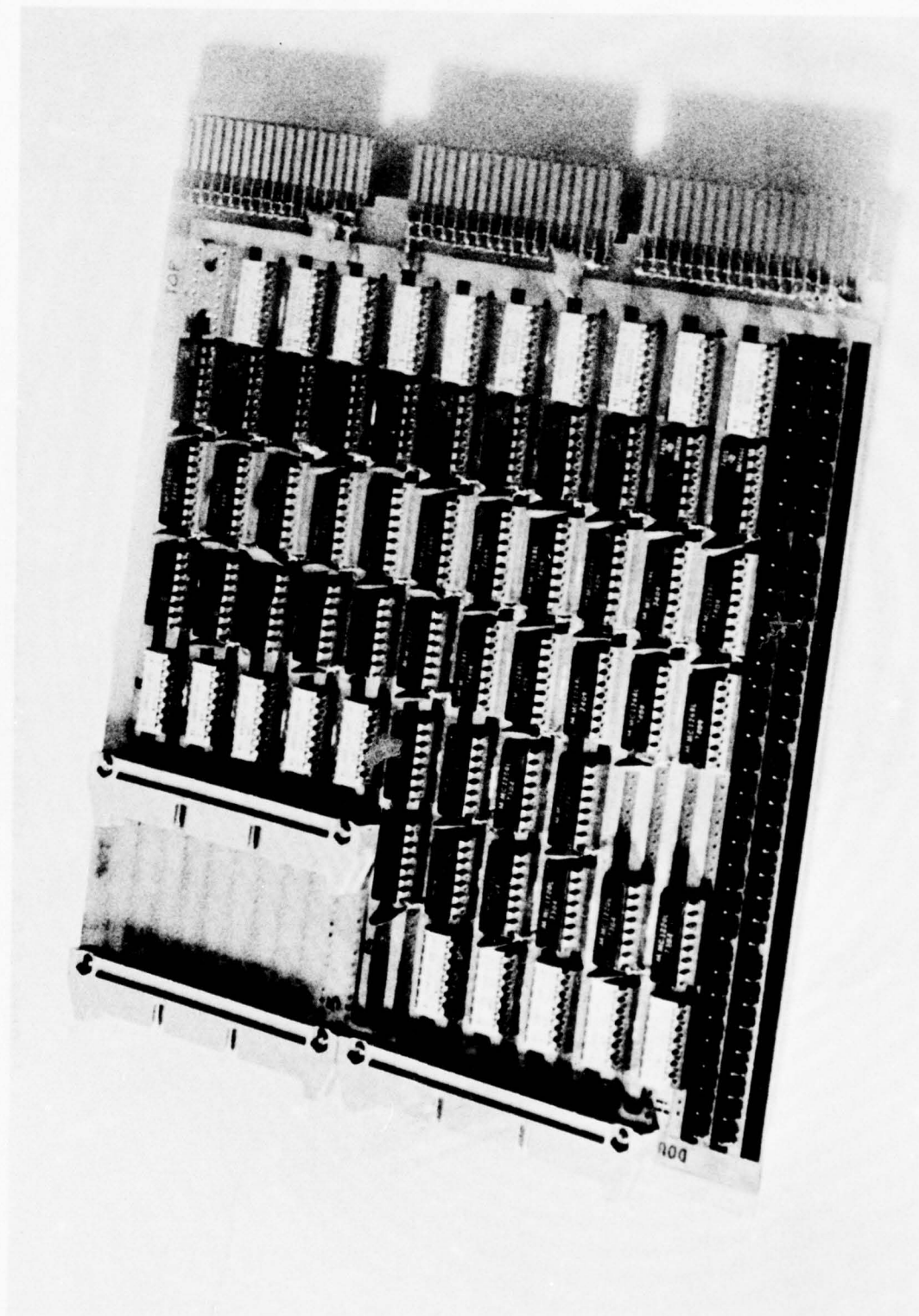


Figure 4.0-7. STARAN Interface Card - Output.

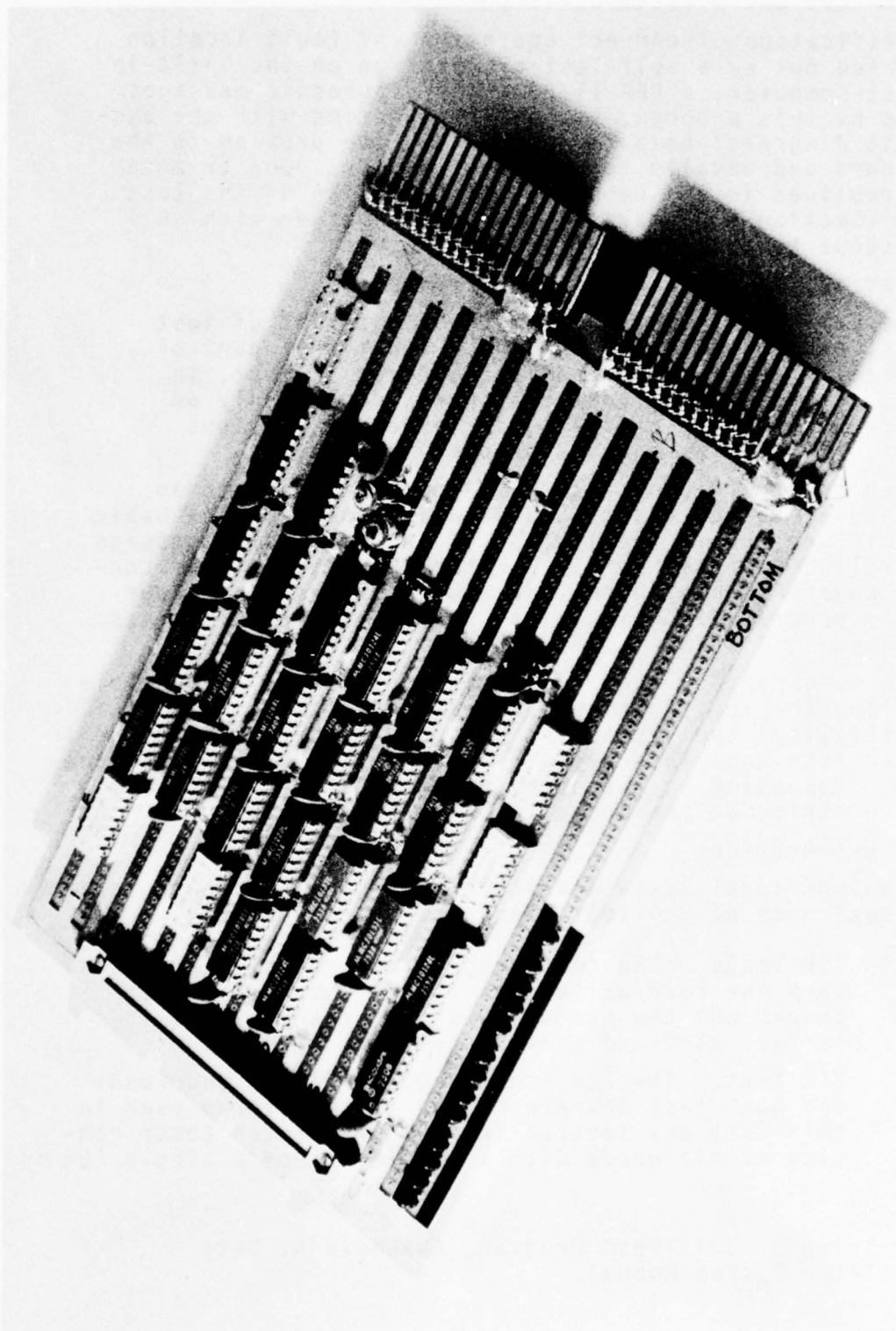


Figure 4.0-8. STARAN Interface Card - Control.



## 5. Self-Test and Maintainability

Verification of correct operation and fault location are carried out by a Self-Test program run on the built-in Self-Test Computer, a PDP-11/03. The diagnostic messages produced by this program, used in conjunction with the system logic diagrams, help to narrow down the problem to the faulty card and usually to within a few ICs. One or more of the routines in the Debugging Aids section of the test program (Section 5.3) may then be used together with an oscilloscope to pinpoint the fault.

### 5.1 Description of the Self-Test Program

The Self-Test program consists of a series of test routines, each of which checks out a functional part of the system. The test routines are ordered so that, in general, each of these functional parts depend only on the correct operation of the logic tested in previous routines, in order to operate properly themselves.

When an error is detected,<sup>1</sup> the program traps to an error handler in the Csect IERR<sup>1</sup>, which generates the basic diagnostic messages. The listing of any particular message is controlled by a bit in the word LMODE. This allows convenient suppression of detailed error messages except during early program debugging, without changing the routines themselves.

The main part of the program in Csect IM is responsible for calling the test routines using a dispatch table. It also initializes the system and responds to Start Test Mode and Clear Test Mode interrupts by stopping and restarting the testing. Depending on the entry point used, it prints summary information at the end of each pass.

#### 5.1.1 Test Routines

The individual tests are performed in the following order during each pass of the test program:

- (i) CSR Tests. The test program writes into and reads back the read/write bits of the CSR. This test checks out the basic operation of the LSI-11 interface card and part of the control card.
- (ii) TIR Test. The TIR is tested by writing and reading back test patterns. The test patterns used in this test and several following register tests consist of all words with a single '1' or a single '0'

---

<sup>1</sup> See Listing of Self-Test Program, Appendix A, Data Manipulator System Manual.

in all positions - the "rippling one" or "bubbling zero" patterns.

- (iii) ACR Tests. Each of the ACR's on the register cards is tested by loading and reading back the rippling one/bubbling zero patterns.
- (iv) Other Registers. Each byte of the other registers on the register cards, ICR, OCR, IMR, OMR, IR, OR and TDR, is tested in the same manner.
- (v) Address Tests. The correctness of the address decoding logic is verified by loading each byte in the registers, ACR, ICR, OCR, IMR, OMR, IR, OR and TDR, and checking if it modified any other byte in any of these registers.
- (vi) STARAN Instruction Tests. The STARAN instructions to load, clear or preset any of the ACRs, ICR, OCR, IMR or OMR are tested for correct operation. The instruction is loaded into the TIR (test instruction register) and for load instructions, a test pattern is loaded into TDR (test data register). The STARAN instruction is executed and the registers are compared against the correct pattern.
- (vii) Data Manipulation Tests. In the first part of this test, the IMR and OMR are set to 1's and the ACR's are loaded with an exhaustive series of patterns that put each multiplexer on the Line Select Cards in all possible settings. For each ACR pattern, the TDR is loaded with a series of rippling 1/ bubbling 0 patterns. For each combination of ACR and TDR patterns, a data manipulation is performed which loads the TDR into the IR and loads the resulting Line Select output into the OR. The OR is then checked against its correct value computed by the program. Approximately 8000 data manipulations are performed during this part of the test.

After that, the data manipulation tests are performed with all the ACRs cleared to zeroes, verifying a correct one-to-all operation.

Lastly, each ACR is set to its address so that each output bit is connected to the corresponding input bit. The TDR is set to 1's. Then ripple one/bubble zero patterns are loaded successively into the IMR or OMR while the other is set to 1's. A data manipu-

lation operation is performed for each IMR or OMR setting and the OR is checked for correctness.

- (viii) ICR/OCR to ACR Conversion Tests. Various patterns are loaded into the ICR and OCR, a conversion operation is performed by issuing a simulated STARAN instruction to perform the conversion. The resulting ACR patterns are compared with values computed by the program, and the ICR is tested to ensure it stayed intact during the conversions. A total of 60 ICR/OCR to ACR operations are performed during each pass.

## 5.2 Running the Data Manipulator Self-Test Program

The Self-Test Program is normally resident in the core memory of the PDP-11/03 microcomputer. An appropriate terminal is connected to the PDP-11/03 through its serial interface connection to communicate with the Self-Test Program. The normal configuration permits terminal speeds of 110 baud or 9600 baud selected by a switch at the end of the serial interface card (DLV-11). These speeds are appropriate for using a teletype or a CRT terminal. Other speeds can be obtained by modifying jumpers on the DLV-11 card. (See LSI-11 PDP-11/03 User's Manual, DEC no. EK-LSI11-TM-001, for details).

When a terminal is connected properly to the PDP-11/03, and the system is powered up but the PDP-11/03 is not running a program, the terminal communicates with the ODT microcode in the computer. In this mode, it responds to a carriage return with a line-feed followed by an @ (at sign).

To start the Self-Test Program at its normal entry point (START), type @20000G.

The program responds by identifying itself:

```
@20000G  
DATA MANIPULATOR SELF-TEST  
-
```

(NOTE: The characters typed by the user are underlined).

The program will continue to run and it will print diagnostic messages if it detects any errors in the hardware. It can be halted at any time by hitting the "break" key. It can also be stopped at any time by a STARAN PIO program that issues the DMC instruction (code FF<sub>16</sub>). It will continue operation upon receipt of a Start Self-Test (code F3<sub>16</sub>) or Resume Self-Test (code F5<sub>16</sub>) instruction from the STARAN.

### 5.2.1 Alternate Entry Points

In addition to the normal entry point (START - address 20000<sub>8</sub>) the following alternate entry points provide variations in the mode of operation:

- (i) CRTREG (address 20604). When started at this address, the program stops after every 20 lines of output; it types a ">" and waits for a carriage return before continuing. When connected to a CRT terminal, this gives the user time to read the screen before it rolls off the top. This mode



could also be useful when the DLV-11 is connected to another computer (e.g., the STARAN sequential processor). This computer could request another buffer-full of data after processing the data in its buffer. The number of lines transmitted each time (and hence the size of the buffer required in the other computer) can be changed by modifying the word at location 20614.

(ii) CRTDTL (address 20624). In this mode, the program provides detailed error diagnostics that are normally suppressed. Basically it prints a diagnostic each time an error is detected whereas in the normal mode all the errors detected during a particular test are often combined and presented as a single meaningful error map. This mode was useful during software debugging and may occasionally be used where the normal error messages don't provide sufficient information. This mode also supports a CRT by stopping after every 20 lines and waiting for a carriage return.

(iii) PASLST (address 31000). When started in this mode, the Self-Test Program prints a pass summary as follows after each pass:

nnnnnn PASSES, mmmmmm WITH ERRORS.

where nnnnnn is an octal number giving the number of passes executed, and mmmmmm (also octal) gives the number of passes during which one or more errors were detected.

(iv) BURNIN (address 20642). This mode is similar to PASLST except that the pass summary is listed every 8 passes (10 octal) instead of each pass. It is useful for system "burn-in" tests.

#### 5.2.2 Format of Diagnostic Messages

The diagnostics printed by the Self-Test Program have the following format:

MSG # nnnnnn

or

MSG # nnnnnn, A = aaaaaa, B = bbbbbb, ....., etc.

where nnnnnn is a number identifying the kind of error detected and aaaaaa, bbbbbb, etc. are octal numbers that provide further details as described under the description for error no. nnnnnn.

e.g.

MSG #16, A = 000210, B = 160403

This message indicates that the program executed a simulated STARAN instruction - Load ICR (code 210<sub>8</sub>) and the resulting ICR was wrong in the byte whose address is 160403 in the PDP-11/03's address space indicating that the problem is on card no. RG4 (see DM Test Computer Address Map - Drawing no. 4051-070676-D).

For certain errors, the diagnostic message of the format described above is followed by other kinds of information. These messages are explained under the description of the relevant diagnostic message.

There is one fatal error that is handled differently. When the program is first started, if the Start Test Interrupt or the Stop Test Interrupt do not function correctly, the program prints

#### PROBLEM WITH TEST MODE INTERRUPT

and halts. This will usually indicate that the +5 volt supply of the data manipulator is off or there is a problem with the LSI-11 interface card.

#### 5.2.3 Error Maps

For certain errors, the standard diagnostic message is followed by an error map which summarizes the results of the particular test in a convenient graphical format. This is presented in the form of a matrix composed of the characters "." (period), "0", "1" and "X". Each position in the matrix represents a bit. A period indicates correct operation, a 0 indicates that a 0 was observed when it should have been a 1, usually a stuck-at-zero situation; a 1 similarly indicates that an incorrect "1" was observed, and an X means both kind of errors were observed for that bit.

#### 5.2.4 Description of Diagnostic Messages

MSG #1: A problem was found in the CSR. Specifically, the program was unable to set and clear CSR bit 3 correctly. A problem on the LSI-11 Interface Card is indicated.

MSG #6: A problem was found while loading and reading back the Test Instruction Register (TIR) on the Control Card. The result of the tests is displayed as a 1x8 error map.

Example:

MSG #6  
....0010

(Bits 0, 2 and 3 appear to be stuck at "0" and bit 1 at "1").

\*MSG #7, A = \_\_, B = \_\_: An error was discovered in loading and reading back a byte in the ACR. The address of the byte is given by the parameter A. The Register Card containing this problem can be deduced as follows: Addresses 16000 to 16007 are on RG1, 160010 to 160017 are on RG2, etc. The second parameter (B) indicates the errors detected: The left half gives the bits found incorrectly set to '1' while the right half gives the bits found incorrectly set to 0.

Example:

MSG #7, A = 160022, B = 000377

(Byte no. 18 (22 octal) of the ACR, which is on card RG3, is stuck at zeroes).

\*MSG #10, A = \_\_, B = \_\_, C = \_\_: An error was found while loading and reading back one of the registers on the Register Cards other than the ACR's.

A gives the Register Card (0=RG1, 1=RG2, etc.)  
B gives the kind of register (0=ICR, 1=OCR, 2=IMR, 3=OMR, 4=IR, 5=OR, 6=TDR); C gives the error pattern observed (see MSG #7).

\*MSG #11, A = \_\_, B = \_\_: Loading a byte of a register whose address in the LSI-11's address space given by B changed the contents of some other byte in some register at the address given by A. This would usually indicate some problem in the addressing part of the self-test circuitry such as a disabled address line.

\* This is a detailed error message; it is suppressed except when the test program is started at CRTDTL.

MSG #12: Loading a byte in a register affected some other byte within the registers of the Data Manipulator. This would usually indicate some problem in the addressing part of the self-test circuitry such as a disabled address line. This message is generated once during a pass at the end of the test for interaction errors of this kind. The test program can be restarted at CRTDTL to get more detailed information (see MSG #11).

MSG #13, A = \_\_, B = \_\_, C = \_\_: A simulated STARAN Instruction to load, clear or preset a bit-slice of the ACR (instructions 1 through 24) did not function properly. A gives the bit of the ACR (0 through 7 - 0 is the LSB), B gives the code in octal of the malfunctioning instruction, C gives the address (in the LSI-11's address space) of the byte where the error was detected.

If no other diagnostics are produced before this during the pass (in particular, MSG #14), the problem would probably lie in the decoder circuitry (such as IC no. N11) on the Register Card involved, which can be obtained from parameter C.

MSG #14: One or more errors were detected while loading the ACR's with different patterns and reading them back. A summary of the errors detected is displayed as an error map following the basic diagnostic message. This is an 8x64 matrix with each row representing the ACRs on a particular Register Card. (See Error Maps, Section 5.2.3).

Examples of this diagnostic are shown in Figure 5.2.4-1. The error map in (a) is annotated to indicate the relationship between the matrix and the logical and physical configuration of the ACRs. Note that bit 0 of any ACR (the right most bit) is the least significant bit.

The error map in (a) shows a problem with bit 1 of ACRs 0 through 3 on Register Card RG4. (ACR24 through 27). Reference to the Register



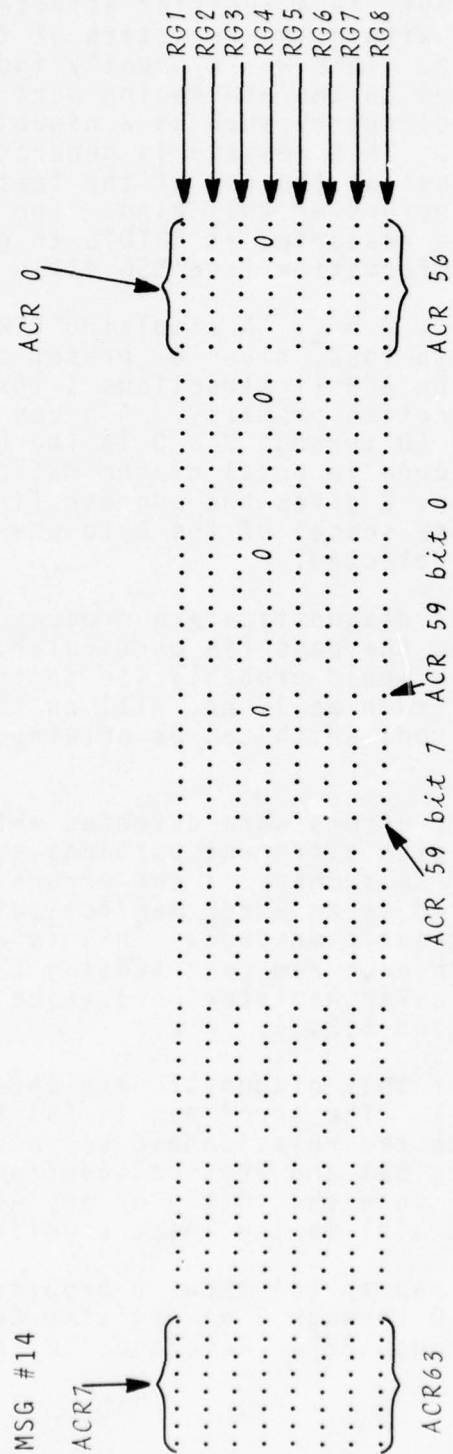


Figure 5.2.4-1(a) - Example of ACR Error Map (MSG #14)

[illegible]

Figure 5.2.4-1(b) - Example of ACR Error Map (MSG #14)

Card logic diagram (Drawing No. 4051-051476-D) points to the multiplexer at IC position B6 as the most likely culprit.

The ACR Error Map in (b) shows a problem in bit 6 of ACR 6 and ACR 7 on Register Card RG7. Again, the logic diagram points to the dual flip-flop (54S74) at IC position H4 as the most likely cause of the problem.

MSG #15: One or more errors were detected while loading one of the 64-bit registers on the Register Cards and reading them back. A summary of the error patterns observed by the program is presented in the form of an error map following the basic diagnostic message.

This is a 7x64 matrix: each row represents a register (ICR, OCR, IMR, OMR, IR, OR, TDR in that order). See Error Maps, Section 5.2.3 for a general description of error maps.

Examples of this diagnostic are shown in Figure 5.2.4-2. The error map in (a) is annotated to indicate the relationship between the matrix displayed, and the logical and physical configuration of the registers.

The error map in (a) shows a problem with bits 0 to 3 of the OMR on RG 6 (OMR bits 40 to 43). Reference to the Register Card logic diagram (Drawing no. 4051-051476-D) indicates that the 54175 at IC location M3 is probably the cause of the problem.

[illegible]



[illegible]

Figure 5.2.4-2(b) - Example of Register Error Map (MSG #15)

The error map in (b) shows a situation where all 8 bits of the ICR on Register Card RG4 are in error. The random 0's and 1's indicate a failure to load. The program probably always read back the same pattern (01100010) irrespective of what it wrote. This suggests the load clock may be absent as a result of a failure in the circuitry associated with decoding the addresses: The problem may be in the IC's at P10, R7 or M1.

MSG #16, A = \_\_, B = \_\_: An error was detected when the test program loaded, preset or cleared one of the 64-bit registers (ICR, OCR, IMR, OMR) by simulating a STARAN instruction.

A gives the instruction code in octal; B gives the address (in the LSI-11 address space) of the byte where the error was detected.

\*MSG#24, A = \_\_, B = \_\_, C = \_\_, D = \_\_, E = \_\_, F = \_\_: An error was detected in the data-manipulation operation. The bit of the OR in error is given by D, and the bit of the IR to which it should have been connected is given by E. Byte number 'A' of the OR should have been 'C' but is 'B'. F gives an internal location where the error is marked, and is useful only for software debugging. A summary of all errors of this type is provided by diagnostic message no. 25.

MSG #25: One or more errors were detected during a test of the data manipulation operation. This is an exhaustive test during which the IMR and OMR are set to 1's, and the ACR's are loaded to make each bit of the OR look at each bit of the IR while a 0 or 1 is rippled through the IR. The result of the test is displayed as an error map that follows the basic diagnostic message. This is in the form of a 64x64 matrix: The rows represent the bits of the IR while the columns represent the bits of the OR. If a 0, 1 or X appears at row i, column j, it means that ACR-j was set at i but OR bit j did not correspond to IR bit i after a data manipulation operation.

\*This is a detailed error message; it is suppressed except when the test program is started at CRTDTL.

Examples of this diagnostic are shown in Figure 5.2.4-3. The error map in (a) is annotated to indicate the relationship between the map elements, and the logical and physical configuration of the Line Select Array and the IR, OR and ACR's.

The row of 0's at A in Figure 5.2.4-3 (a) indicates a possible problem with bit 4 of the IR, IMR or the related gates on the Register Card RG3 since the output is zero whenever any of the multiplexers looks at IR bit 20. The 6 rows of 1's at B indicate problems with the 6 input signals on Line Select card LS1 for inputs 37 through 42. Since these six signals pass through the hex-receiver (7837) at IC position 63 (refer to the logic diagram of the Line Select Card - Drawing no. 4051-070876-D<sup>1</sup>), this IC is probably at fault.

The X's at A in Figure 5.2.4-3(b) result from a problem on one of the ACR lines. The multiplexer for bit 2 on card LS1 gives a wrong output whenever it is addressed with an odd number. This would happen if bit 0 of ACR2 on Register Card RG1 was not reaching bit 0 of the corresponding multiplexer on card LS1, probably because of a problem in the receiver (7837) at IC position 108<sup>1</sup>. The column of 1's at B means that bit 39 of the OR is always a '1', irrespective of the contents of ACR 39 or the corresponding input bit. This could happen because of a defective multiplexer whose output is always low, indicating that one of the 7219's at IC locations 8, 30, 52 or 74<sup>1</sup> is defective. At C, we see the Error Map pattern produced by a short circuit between the two signals corresponding to input bits 9 and 11 on card LS4.

MSG #26: An error occurred in the data manipulation operation when all the ACR's were set to 0.

MSG #27, A=\_, B=\_: A problem was discovered during the test of the input and output masking operation. In this test, the IR is set to 1's, and the ACRs are loaded with their addresses, i.e., to pass data straight through. A single 0 or 1 is then rippled through either the IMR or the OMR, and the OR is checked for corrections. If there is no previous error message,

<sup>1</sup> Data Manipulation - Engineering Drawings, Drawing no. 4051-070876-D.

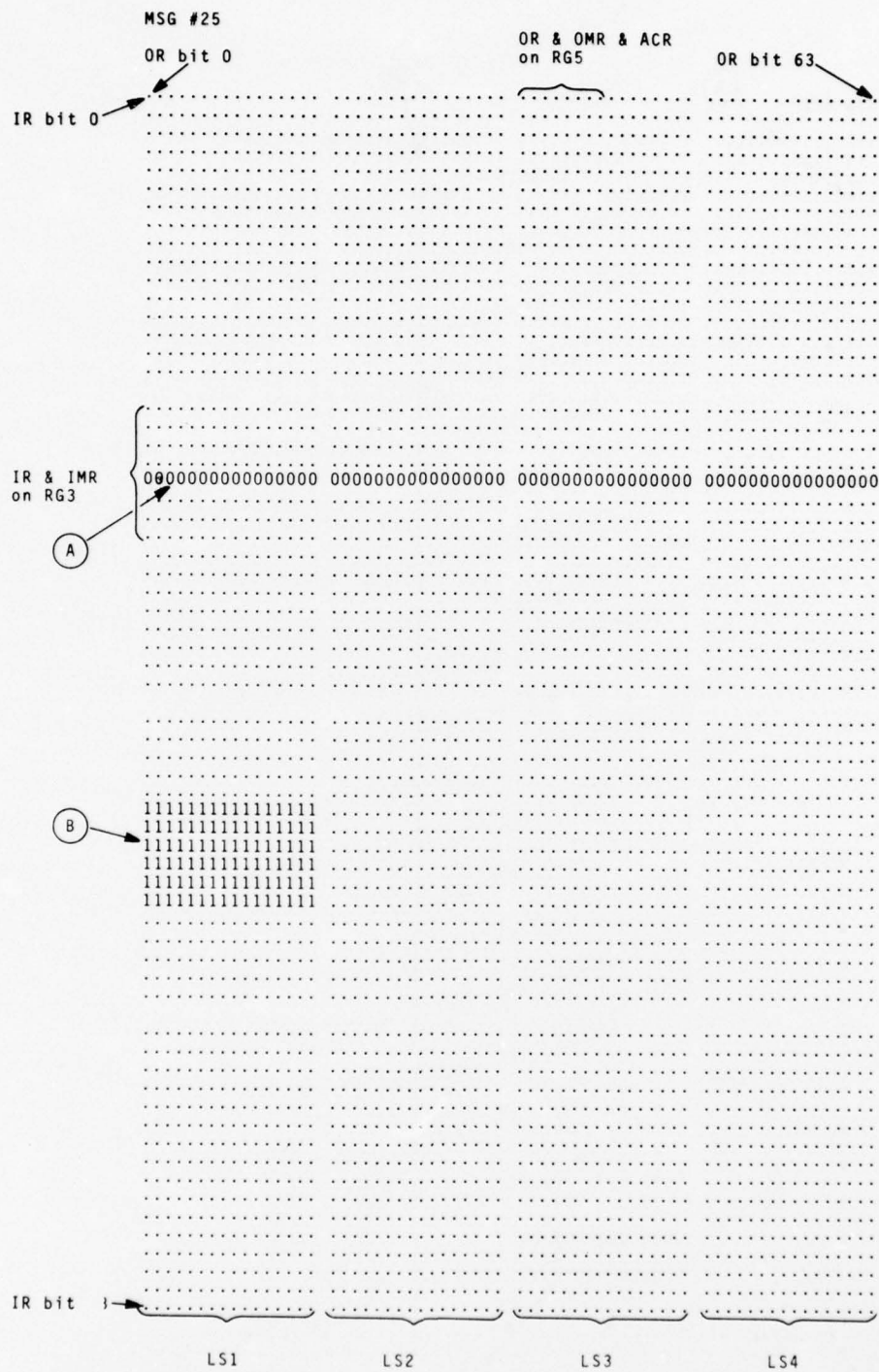


Figure 5.2.4-3(a) - Example of Data Manipulation Error Map (MSG #25)



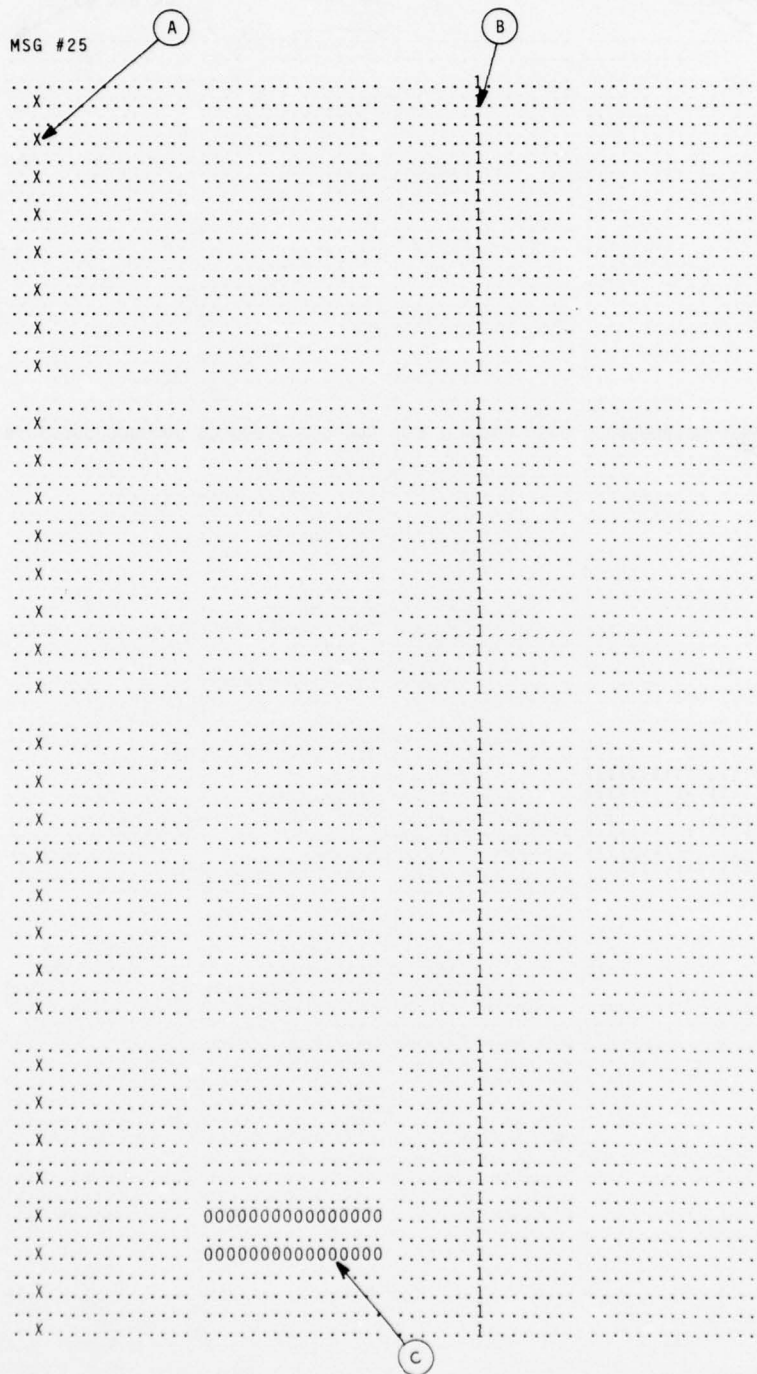


Figure 5.2.4-3(b) - Example of Data Manipulation Error Map  
(MSG #25)

an error in the masking operation is indicated. A gives the Register Card on which the error is found (0=RG1, 1=RG2, etc.) and B gives the address of the byte of the Mask Register where the problem was found.

MSG #30: A problem occurred in the ICR/OCR to ACR conversion. The message is followed by a binary listing of the ICR and OCR, the resulting ACRs and the correct ACR pattern as computed by the program. An example of such a display is shown in Figure 5.2.4-4. In this example, the problem is seen to be on RG8.

MSG #31: The ICR got modified during an ICR/OCR conversion operation. A binary representation of the ICR before and after the conversion is printed following the basic diagnostic message. An example of this message is shown in Figure 5.2.4-5.

The part of the ICR on RG6 is seen to have changed during the conversion.

\*MSG#36, A = \_\_, B = \_\_: The stack-pointer was modified by some test routine indicating a software bug. A = SP after the routine, B = address of routine entry in table SUBLST (value of TSTPTR).

MSG #37: A time-out trap occurred. This could be due to a problem on the LSI-Interface Card resulting in an illegal memory reference, or due to a software error.

\* This is a detailed error message; it is suppressed except when the test program is started at CRTDTL.

```

MSG #30
10101010 10101010 10101010 10101010 10101010 10101010 10101010 10101010
10101010 10101010 10101010 10101010 10101010 10101010 10101010 10101010
00000111 00000000 00000101 00000000 00000011 00000000 00000001 00000000
00001111 00000000 00001101 00000000 00001011 00000000 00001001 00000000
00010111 00000000 00010101 00000000 00010011 00000000 00010001 00000000
00011111 00000000 00011101 00000000 00011011 00000000 00011001 00000000
00100111 00000000 00100101 00000000 00100011 00000000 00100001 00000000
00101111 00000000 00101101 00000000 00101011 00000000 00101001 00000000
00110111 00000000 00110101 00000000 00110011 00000000 00110001 00000000
00111101 00000000 00111101 00000000 00111001 00000000 00000000 00000000
00000111 00000000 00000101 00000000 00000011 00000000 00000001 00000000
00001111 00000000 00001101 00000000 00001011 00000000 00001001 00000000
00010111 00000000 00010101 00000000 00010011 00000000 00010001 00000000
00011111 00000000 00011101 00000000 00011011 00000000 00011001 00000000
00100111 00000000 00100101 00000000 00100011 00000000 00100001 00000000
00101111 00000000 00101101 00000000 00101011 00000000 00101001 00000000
00110111 00000000 00110101 00000000 00110011 00000000 00110001 00000000
00111111 00000000 00111101 00000000 00111011 00000000 00111001 00000000

```

Figure 5.2.4-4 - Example of ICR/OCR Conversion Error display.

MSG #31

10101010 10101010 10101010 10101010 10101010 10101010 10101010  
10101010 10101010 10101110 10101010 10101010 10101010 10101010

Figure 5.2.4-5. Example of ICR Modification error display.



### 5.3 Debugging Aids

The Self-Test Program includes a number of routines to aid in debugging the Data Manipulator. These routines appear in the Csect IDEB.

- (i) TST8 (address - 27154). This routine tests 8 adjacent bytes in the LSI-11's address space by writing and reading back a number of patterns (rippling one/bubbling zero) for each of these bytes. The address of the first byte in octal must be typed in in response to a "?" from the program. The result is displayed as 8 16-bit words in binary. The left byte indicates the bits read back erroneously as 1's while the right byte gives the bits read back erroneously as 0's. For correct operation, the result would consist of all 0's. Typing a "P" causes the routine to test the next 8 bytes.
- (ii) TST1 (address 27236). Tests a single byte. See TST8 above.
- (iii) DISP (address 27250). The state of all the registers of the Data Manipulator is displayed in binary. Figure 5.3-1 shows the relationship between the display and the register bits, along with their physical location.
- (iv) CLACR (address 27274). The ACRs are cleared, followed by a call to DISP.
- (v) LDACR (address 27276). The ACRs are all loaded with a specified pattern from R2, followed by a call to DISP.
- (vi) SETACR (address 27324). The ACRs are set to 1's, followed by a call to DISP.
- (vii) CON (address 27332). An ICR/OCR to ACR conversion is performed and the result is displayed by a call to DISP.
- (viii) CONC (address 27356). ICR/OCR to ACR conversions are performed repetitively, to facilitate debugging with an oscilloscope.
- (ix) SIX (address 27402). The execution of STARAN Instruction for the Data Manipulator is simulated and the result is displayed by a call to DISP.

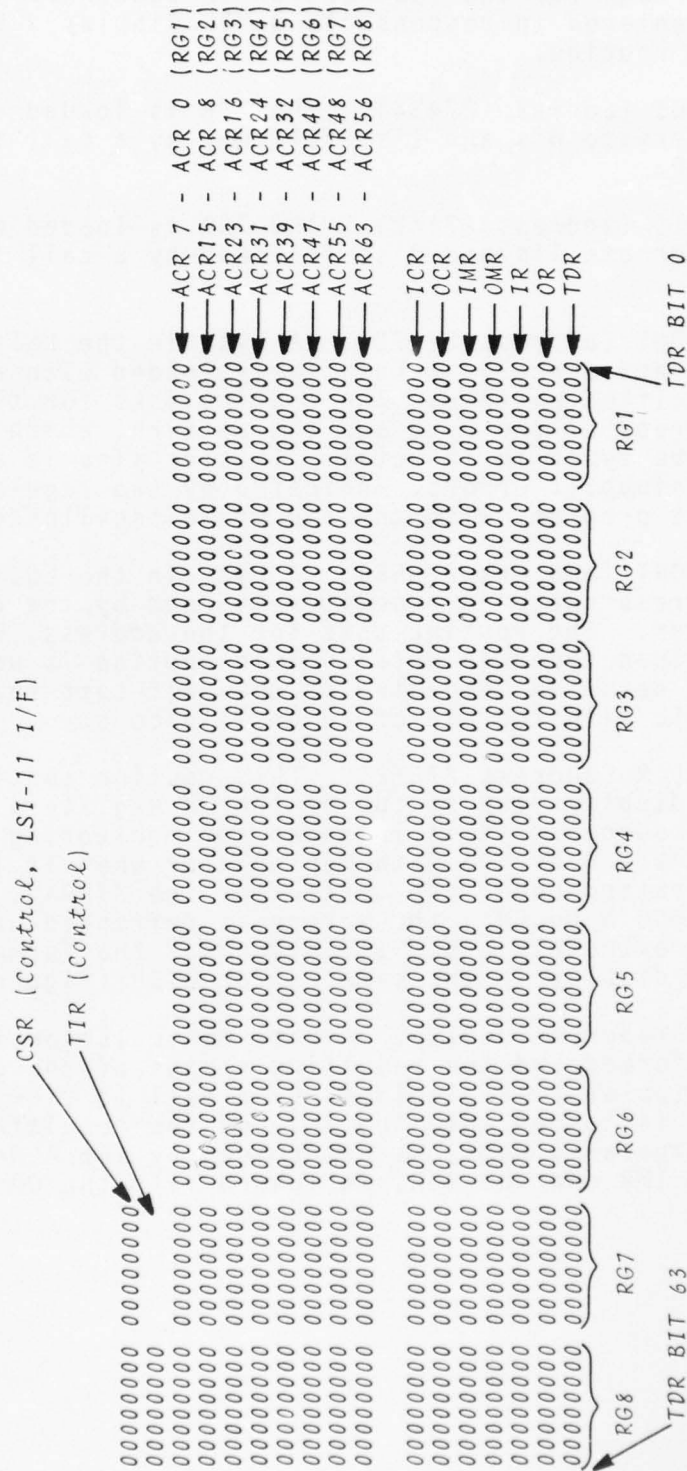


Figure 5.3-1 - Format of DM Register Display (generated by DISP, MONITR, etc.)

The code for the instruction in octal has to be entered in response to a "?" displayed by the routine.

- (x) ALT01 (address 27434). The TDR is loaded with alternate 0's and 1's, followed by a call to DISP.
- (xi) ALT10 (address 27442). The TDR is loaded with alternate 1's and 0's, followed by a call to DISP.
- (xii) LDCONT (address 27472). A byte in the LSI-11's address space is repetitively loaded with a specified pattern. The routine asks for the address of the byte and the pattern, which have to be typed in in octal. This routine is used to pinpoint errors, indicated by the regular test program, with the aid of an oscilloscope.
- (xiii) RDCONT (address 27556). A byte in the LSI-11's address space is repetitively read by the computer. The routine asks for the address, which is then typed in octal. This routine is used for detailed debugging of the self-test read logic with the aid of an oscilloscope.
- (xiv) MONITR (address 27612). This routine continuously displays the status of the DM Registers on a CRT screen. It is intended for monitoring the operation of the Data Manipulator when it is operating under the control of the STARAN. With a 9600 baud CRT, the screen is refreshed once approximately every 1.2 seconds. The format of the display is the same as for DISP (Figure 5.3-1).
- (xv) DMX (address 30012). A data manipulation is performed and the resulting status of the Data Manipulator is displayed by a call to DISP. The TDR is loaded into the IR, and the resulting manipulated data, as determined by the ACRs, the IMR and the OMR, is loaded into the OR.

## 6. INSTALLATION AND OPERATION OF THE DATA MANIPULATOR IN THE RADC PARALLEL/ASSOCIATIVE COMPUTER FACILITY

The connectors, cables and interface circuitry (ECL-to-TTL conversion) were designed and constructed early in the program, within a few months after award. They were then installed in the STARAN Computer and all signal levels and their timing, including clock timing, were carefully verified. The cables and connectors were then stored in the STARAN Computer cabinets and underneath the floor of the computer room, ready for final installation of the Data Manipulator. Even though the STARAN Computer was moved to a different location between initial interface testing and the final Data-Manipulator installation, no problems were encountered with this procedure.

After construction of the Data Manipulator at the laboratories of W. W. Gaertner Research, Inc. in Stamford, Connecticut, the circuit cards, power supplies, fans and the PDP-11/03 were removed from the cabinet prior to shipping. After arrival at RADC, the entire Data Manipulator cabinet was reassembled in less than one day. Systematic testing began immediately, using, initially, primarily the Self-Test Computer and both a Decscope and teletype terminals. After initial verification of correct operation, overnight life tests were conducted, logging all transient or permanent errors. Since the Self-Test program performs over 32 million fully verified data manipulations in 15 hours, such overnight tests verify not only the correct operation of all functions but also the transient error rate of the Data Manipulator.

After successful completion of all initial tests with the built-in Self-Test program, systematic testing with the STARAN Computer was carried out.

This was implemented by using the STARAN Control Module (SCM) and the STARAN Debug Module (SDM). They enable the operator to directly issue External Function (EXF) Commands to any element of STARAN, inspect and change any memory location in the Parallel I/O (PIO) Control Memory, start and halt PIO Instructions in the PIO Control Memory, as well as many other operations via the Keyboard/Printer Control Console.

Programs were written and executed to verify the following operations between the Data Manipulator (DM) and the STARAN Computer:

writing of data to DM from PIO Control Memory,



reading of data from DM into PIO Control Memory,  
 swapping of array memories with DM,  
 testing the various clock pulses to DM,  
 initialize all DM registers,  
 sending the various control signals to and from  
 DM, and  
 performing the various data manipulating operations.

Due to the present lack of higher-level software for  
 the Custom I/O Unit (CIOU) of the STARAN Computer, the test  
 programs had to be written in machine code. The following  
 example is a short program used for initializing the DM  
 registers:

<u>PIO Control Memory Location</u>	<u>Code</u>	<u>Comment</u>
0A00	2801 0A01	unconditional branch to memory location 0A01
0A01	3641 0A10	BCW1 and BCW2 are given in 0A10
0A02	40FF 0844	enter DM Control Mode
0A03	4082 0844	preset ICR
0A04	4092 0844	preset OCR
0A05	40A2 0844	preset IMR
0A06	40B2 0844	preset OMR
0A07	40F9 0844	perform ICR/OCR-ACR Conversion and enter DM mode
0A08	3801 8000	halt PIO operation
0A10	0843 0843	BCW1 and BCW2

A detailed description of the SCM, SDM and the operation  
 of the PIO can be found in the following manuals:

STARAN User's Guide  
GER-15644C  
Goodyear Aerospace Corporation,

STARAN CIOU Reference Manual  
GER-156-42  
Goodyear Aerospace Corporation.

RADC in-house personnel are currently developing macros for the STARAN Computer which can be readily called by other programs and which utilize all capabilities of the Data Manipulator.

Since the Data Manipulator is powered up together with the STARAN Computer at all times, and connected to the ARPANET, it is rapidly accumulating very substantial numbers of operating hours.

## 7. CONCLUSIONS AND OUTLOOK

The conclusions can be drawn from this project that a Data Manipulator provides a very useful building block for a variety of computer architectures. While there has previously been strong hesitation to implement massive cross-point switches (256x256 lines require 65,536 cross-over points), modern off-the-shelf MSI circuitry makes the construction of such a Data Manipulator quite straightforward. A propagation delay of less than 250 ns (i.e. less than one-third of the original specification) can be easily achieved which makes the Data Manipulator constructed by W. W. Gaertner Research, Inc. readily speed-compatible with the STARAN Computer and most other standard computer systems. Using a built-in mini/microcomputer to provide comprehensive self-test and diagnostic capability proved an invaluable tool for initial debugging, demonstration testing and ultimate equipment maintainability. The fact that the Data Manipulator can be operated even if disconnected from the STARAN Computer allows ready separation of fault causes between the 2 computer systems, and it made it unnecessary to develop comprehensive test and diagnostic software in the STARAN Computer simply to provide maintainability for the Data Manipulator.

While tailored to the STARAN Computer, the Data Manipulator design which was developed under this Contract could be readily adapted to a wide range of other computer and communication systems. Considering the ease with which it was integrated with a STARAN Computer that had been installed several years earlier, there appears to be little risk in retrofitting a Data Manipulator onto other existing computer systems.

It is hoped that this program has shown both the desirability and the feasibility of incorporating a highly versatile Data-Manipulating function into present and future computer architectures.

# METRIC SYSTEM

## BASE UNITS:

Quantity	Unit	SI Symbol	Formula
length	metre	m	...
mass	kilogram	kg	...
time	second	s	...
electric current	ampere	A	...
thermodynamic temperature	kelvin	K	...
amount of substance	mole	mol	...
luminous intensity	candela	cd	...

## SUPPLEMENTARY UNITS:

plane angle	radian	rad	...
solid angle	steradian	sr	...

## DERIVED UNITS:

Acceleration	metre per second squared	...	m/s
activity (of a radioactive source)	disintegration per second	...	(disintegration)/s
angular acceleration	radian per second squared	...	rad/s
angular velocity	radian per second	...	rad/s
area	square metre	...	m
density	kilogram per cubic metre	...	kg/m
electric capacitance	farad	F	A·s/V
electrical conductance	siemens	S	A/V
electric field strength	volt per metre	...	V/m
electric inductance	henry	H	V·s/A
electric potential difference	volt	V	W/A
electric resistance	ohm	...	V/A
electromotive force	volt	V	W/A
energy	joule	J	N·m
entropy	joule per kelvin	...	J/K
force	newton	N	kg·m/s
frequency	hertz	Hz	(cycle)/s
illuminance	lux	lx	lm/m
luminance	candela per square metre	...	cd/m
luminous flux	lumen	lm	cd·sr
magnetic field strength	ampere per metre	...	A/m
magnetic flux	weber	Wb	V·s
magnetic flux density	tesla	T	Wb/m
magnetomotive force	ampere	A	...
power	watt	W	J/s
pressure	pascal	Pa	N/m
quantity of electricity	coulomb	C	A·s
quantity of heat	joule	J	N·m
radiant intensity	watt per steradian	...	W/sr
specific heat	joule per kilogram-kelvin	...	J/kg·K
stress	pascal	Pa	N/m
thermal conductivity	watt per metre-kelvin	...	W/m·K
velocity	metre per second	...	m/s
viscosity, dynamic	pascal-second	...	Pa·s
viscosity, kinematic	square metre per second	...	m/s
voltage	volt	V	W/A
volume	cubic metre	...	m
wavenumber	reciprocal metre	...	(wave)/m
work	joule	J	N·m

## SI PREFIXES:

Multiplication Factors	Prefix	SI Symbol
1 000 000 000 000 = 10 <sup>12</sup>	tera	T
1 000 000 000 = 10 <sup>9</sup>	giga	G
1 000 000 = 10 <sup>6</sup>	mega	M
1 000 = 10 <sup>3</sup>	kilo	k
100 = 10 <sup>2</sup>	hecto*	h
10 = 10 <sup>1</sup>	deka*	da
0.1 = 10 <sup>-1</sup>	deci*	d
0.01 = 10 <sup>-2</sup>	centi*	c
0.001 = 10 <sup>-3</sup>	milli	m
0.000 001 = 10 <sup>-6</sup>	micro	μ
0.000 000 001 = 10 <sup>-9</sup>	nano	n
0.000 000 000 001 = 10 <sup>-12</sup>	pico	p
0.000 000 000 000 001 = 10 <sup>-15</sup>	femto	f
0.000 000 000 000 000 001 = 10 <sup>-18</sup>	atto	a

\* To be avoided where possible.



*MISSION  
of  
Rome Air Development Center*

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C<sup>3</sup>) activities, and in the C<sup>3</sup> areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

